Raport 3

Paweł Matławski
album 249732

13 marca 2021

Spis treści

1 Klasyfikacja na bazie modelu regresji liniowej

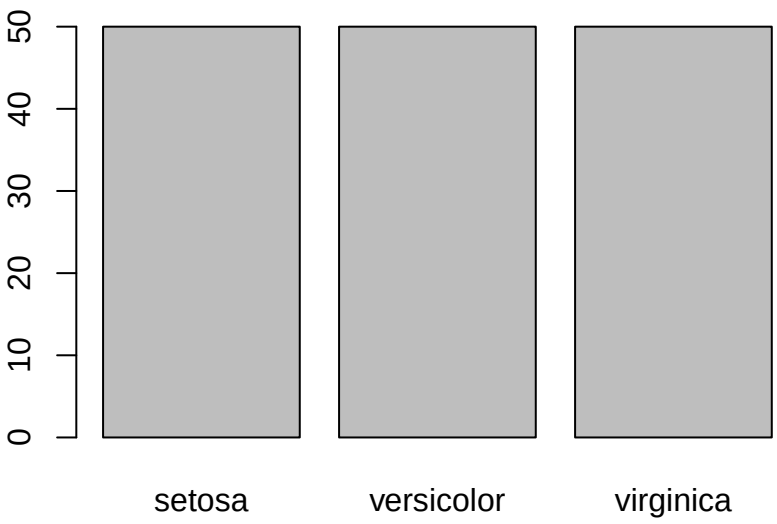
1.1 Wybór danych

Zadanie wykonamy dla ramki danych **iris** z pakietu **datasets**. Przyjrzyjmy się jej:

```
library("datasets")
data("iris")
attach(iris)

#liczba kolumn
ncol(iris)

## [1] 5

#liczba wierszy
nrow(iris)

## [1] 150
```

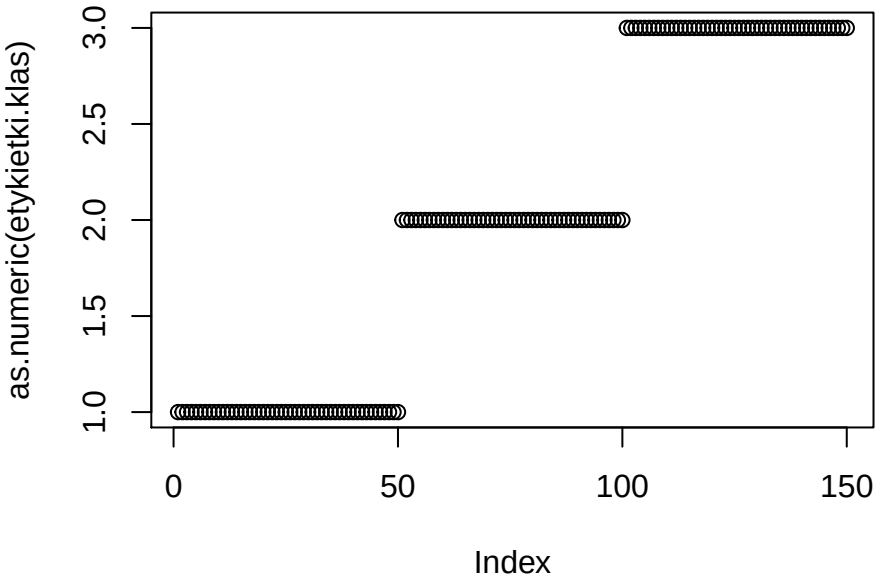
```r
#brakujące dane
sum(is.na(iris))

## [1] 0

#etykietki klas
etykietki.klas <- iris$Species

#liczba obiektów
(n <- length(etykietki.klas))

## [1] 150

#liczba klas
(K <- length(levels(etykietki.klas)))

## [1] 3
```

```r
plot(etykietki.klas)
```



Rysunek 1: Przynależność do poszczególnych gatunków

```r
plot(as.numeric(etykietki.klas))
```



Rysunek 2: Przynależność do poszczególnych gatunków

Wnioski:

- Obiekty są uporządkowane ze względu na przynależność do poszczególnych gatunków - po 50 z każdego.
- Mamy 3 klasy i 150 obiektów.

1.2 Podział danych na zbiór uczący i testowy

Podzielimy dane w proporcji:

- $\frac{2}{3}$-zbiór uczący
- $\frac{1}{3}$-zbiór testowy.

```
#losowanie obiektów
learning.indx <- sample(1:n,2/3*n)

#zbiór uczący
learning.set <- iris[learning.indx,]

#zbiór testowy
test.set <- iris[-learning.indx,]
```

1.3 Konstrukcja klasyfikatora i wyznaczenie prognoz

```
#etykietki klas
etykietki.klas.ucz <- learning.set$Species
etykietki.klas.test <- test.set$Species

#liczba obiektów
n.ucz <- length(etykietki.klas.ucz)
n.test <- length(etykietki.klas.test)

#liczba klas
k.ucz <- length(levels(etykietki.klas.ucz))
k.test <- length(levels(etykietki.klas.test))
```

Wyznaczamy macierze eksperymentu *X.ucz* oraz *X.test*, które zawierają wartości poszczególnych zmiennych. W pierwszych kolumnach umieszczamy jedynki, by uwzględnić wyrazy wolne:

```
X.ucz <- cbind(rep(1,100), learning.set[,1:4])
X.ucz <- as.matrix(X.ucz)
X.test <- cbind(rep(1,50), test.set[,1:4])
X.test <- as.matrix(X.test)
```

```
print(xtable(head(X.ucz), caption="Początkowy fragment macierzy X.ucz"))
```

	rep(1, 100)	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
148	1.00	6.50	3.00	5.20	2.00
93	1.00	5.80	2.60	4.00	1.20
105	1.00	6.50	3.00	5.80	2.20
138	1.00	6.40	3.10	5.50	1.80
95	1.00	5.60	2.70	4.20	1.30
127	1.00	6.20	2.80	4.80	1.80

Tabela 1: Początkowy fragment macierzy X.ucz

```
print(xtable(head(X.test), caption="Początkowy fragment macierzy X.test"))
```

Tworzymy macierze *Y.ucz* oraz *Y.test*, które zawierają zmienne binarne kodujące poszczególne klasy.

```
Y.ucz <- matrix(0, nrow=n.ucz, ncol=k.ucz)
Y.test <- matrix(0, nrow=n.test, ncol=k.test)

etykietki.num.ucz <- as.numeric(etykietki.klas.ucz)
etykietki.num.test <- as.numeric(etykietki.klas.test)
```

	rep(1, 50)	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	1.00	5.10	3.50	1.40	0.20
3	1.00	4.70	3.20	1.30	0.20
4	1.00	4.60	3.10	1.50	0.20
7	1.00	4.60	3.40	1.40	0.30
8	1.00	5.00	3.40	1.50	0.20
12	1.00	4.80	3.40	1.60	0.20

Tabela 2: Początkowy fragment macierzy X.test

```
for (k in 1:k.ucz)
  Y.ucz[etykietki.num.ucz==k, k] <- 1
for (k in 1:k.test)
  Y.test[etykietki.num.test==k, k] <- 1
```

Następnie wyznaczymy prognozowane prawdopodobieństwa przynależności do poszczególnych klas.

```
# Macierz estymowanych współczynników
B <- solve(t(X.ucz)%*%X.ucz) %*% t(X.ucz) %*% Y.ucz

# Prognozowane prawdopodobieństwa przynależności do poszczególnych klas
Y.ucz.hat <- X.ucz%*%B
Y.test.hat <- X.test%*%B
```

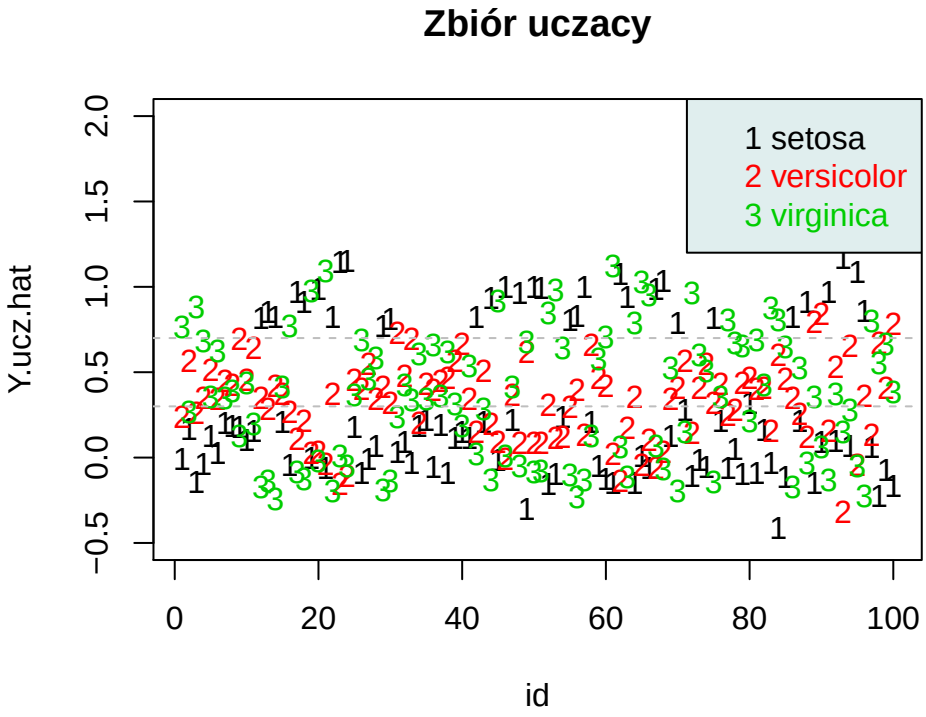
```
print(xtable(head(Y.ucz.hat), caption="Wartości prognozowane"))
```

	1	2	3
148	-0.00	0.24	0.76
93	0.17	0.57	0.26
105	-0.14	0.26	0.88
138	-0.04	0.35	0.68
95	0.13	0.52	0.35
127	0.03	0.35	0.62

Tabela 3: Wartości prognozowane

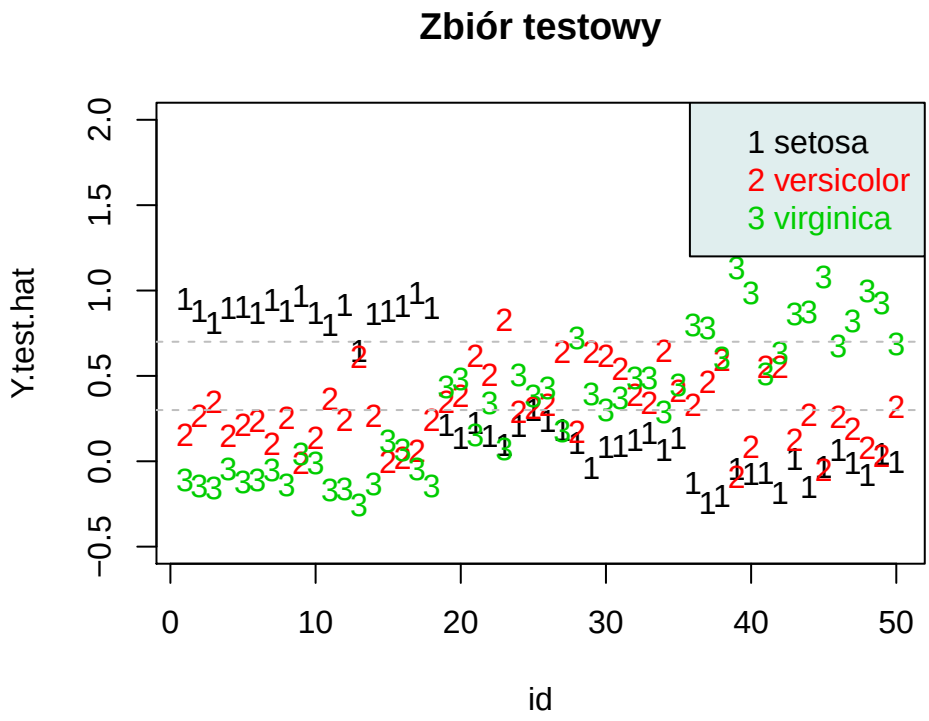
Graficzna prezentacja wyników dla obu zbiorów:

```
par(mar = c(4, 4, 4, 1))
matplot(Y.ucz.hat, main="Zbiór uczący",xlab="id", ylim=c(-.5,2))
abline(h=c(0.3,0.7), lty=2, col="gray")
legend(x="topright", legend=paste(1:3,levels(learning.set$Species)), col=1:3, text.col=1
```



Rysunek 3: Prognozy dla zbioru uczącego

```
par(mar = c(4, 4, 4, 1))
matplot(Y.test.hat, main="Zbiór testowy",xlab="id", ylim=c(-.5,2))
abline(h=c(0.3,0.7), lty=2, col="gray")
legend(x="topright", legend=paste(1:3,levels(test.set$Species)), col=1:3, text.col=1:3,
```



Rysunek 4: Prognozy dla zbioru testowego

1.4 Ocena jakości modelu

```
klasy.ucz <- levels(learning.set$Species)
klasy.test <- levels(test.set$Species
                     )
maks.ind.ucz <- apply(Y.ucz.hat, 1, FUN=function(x) which.max(x))
maks.ind.test <- apply(Y.test.hat, 1, FUN=function(x) which.max(x))

prognozowane.etykietki.ucz <- klasy.ucz[maks.ind.ucz]
prognozowane.etykietki.test <- klasy.test[maks.ind.test]

rzeczywiste.etykietki.ucz <- etykietki.klas.ucz
rzeczywiste.etykietki.test <- etykietki.klas.test

# macierze pomyłek
macierz.pomylek.ucz <- table(rzeczywiste.etykietki.ucz, prognozowane.etykietki.ucz)

macierz.pomylek.test <- table(rzeczywiste.etykietki.test, prognozowane.etykietki.test)
```

```
print(xtable(macierz.pomylek.ucz, caption="Macierz pomyłek - zbiór uczący"))
```

	setosa	versicolor	virginica
setosa	32	0	0
versicolor	0	24	9
virginica	0	4	31

Tabela 4: Macierz pomyłek - zbiór uczący

```
# dokładność klasyfikacji - zbiór uczący
sum(diag(macierz.pomylek.ucz))/n.ucz

## [1] 0.87
```

```
print(xtable(macierz.pomylek.test, caption="Macierz pomyłek - zbiór testowy"))
```

	setosa	versicolor	virginica
setosa	18	0	0
versicolor	0	8	9
virginica	0	1	14

Tabela 5: Macierz pomyłek - zbiór testowy

```
# dokładność klasyfikacji - zbiór testowy
sum(diag(macierz.pomylek.test))/n.test

## [1] 0.8
```

Wnioski:

- Dokładność klasyfikacji stoi na przyzwoitym poziomie, choć z pewnością wciąż jest miejsce na poprawę.
- Gatunek setosa jest dobrze wydzielony od reszty, co widać zarówno na wykresie, jak i w macierzy pomyłek.
- W tym przypadku występuje problem maskowania klas, o którym była mowa na wykładzie.

1.5 Budowa modelu liniowego dla rozszerzonej przestrzeni cech

Tworzymy nowy model regresji uzupełniony o składniki wielomianowe stopnia 2 (tzn. $PL^2, PW^2, SL^2, SW^2, PL * PW, PL * SW, PL * SL, PW * SL, PW * SW, SL * SW$). Powtarzamy wszystkie kroki tak, jak w poprzednim przypadku.

```
iris3 <- iris[,1:4]
names(iris3)[1] <- "SL"
names(iris3)[2] <- "SW"
names(iris3)[3] <- "PL"
```

```
names(iris3)[4] <- "PW"

iris2 <- iris
Species <- iris2$Species
iris2 <- iris2[,1:4]


iris2$Sepal.Length <- iris2$Sepal.Length*iris2$Sepal.Length
iris2$Sepal.Width <- iris2$Sepal.Width*iris2$Sepal.Width
iris2$Petal.Length <- iris2$Petal.Length*iris2$Petal.Length
iris2$Petal.Width <- iris2$Petal.Width*iris2$Petal.Width

names(iris2)[1] <- "SL^2"
names(iris2)[2] <- "SW^2"
names(iris2)[3] <- "PL^2"
names(iris2)[4] <- "PW^2"
attach(iris2)

SL.SW <- iris$Sepal.Length*iris$Sepal.Width
SL.PL <- iris$Sepal.Length*iris$Petal.Length
SL.PW <- iris$Sepal.Length*iris$Petal.Width
SW.PL <- iris$Sepal.Width*iris$Petal.Length
SW.PW <- iris$Sepal.Width*iris$Petal.Width
PL.PW <- iris$Petal.Length*iris$Petal.Width

iris.new <- cbind(iris3,iris2, SL.SW, SL.PL, SL.PW, SW.PL, SW.PW, PL.PW, Species)
```

```
n.new <- dim(iris.new)[1]

#losowanie obiektów
learning.indx.new <- sample(1:n.new,2/3*n.new)

#zbiór uczący
learning.set.new <- iris.new[learning.indx.new,]

#zbiór testowy
test.set.new <- iris.new[-learning.indx.new,]

#etykietki klas
etykietki.klas.ucz.new <- learning.set.new$Species
etykietki.klas.test.new <- test.set.new$Species

#liczba obiektów
n.ucz.new <- length(etykietki.klas.ucz.new)
n.test.new <- length(etykietki.klas.test.new)

#liczba klas
```

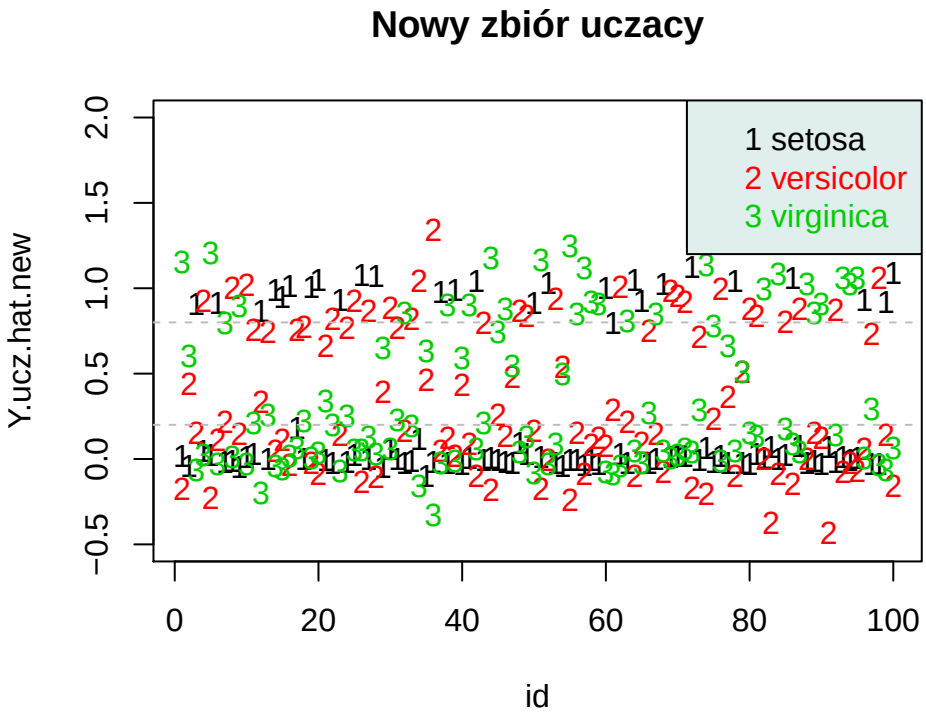
```r
k.ucz.new <- length(levels(etykietki.klas.ucz.new))
k.test.new <- length(levels(etykietki.klas.test.new))

X.ucz.new <- cbind(rep(1,100), learning.set.new[,1:14])
X.ucz.new <- as.matrix(X.ucz.new)
X.test.new <- cbind(rep(1,50), test.set.new[,1:14])
X.test.new <- as.matrix(X.test.new)

Y.ucz.new <- matrix(0, nrow=n.ucz.new, ncol=k.ucz.new)
Y.test.new <- matrix(0, nrow=n.test.new, ncol=k.test.new)

etykietki.num.ucz.new <- as.numeric(etykietki.klas.ucz.new)
etykietki.num.test.new <- as.numeric(etykietki.klas.test.new)

for (k in 1:k.ucz.new)
  Y.ucz.new[etykietki.num.ucz.new==k, k] <- 1
for (k in 1:k.test.new)
  Y.test.new[etykietki.num.test.new==k, k] <- 1

# Macierz estymowanych współczynników
B.new <- solve(t(X.ucz.new)%*%X.ucz.new) %*% t(X.ucz.new) %*% Y.ucz.new

# Prognozowane prawdopodobieństwa przynależności do poszczególnych klas
Y.ucz.hat.new <- X.ucz.new%*%B.new
Y.test.hat.new <- X.test.new%*%B.new

klasy.ucz.new <- levels(learning.set.new$Species)
klasy.test.new <- levels(test.set.new$Species)

maks.ind.ucz.new <- apply(Y.ucz.hat.new, 1, FUN=function(x) which.max(x))
maks.ind.test.new <- apply(Y.test.hat.new, 1, FUN=function(x) which.max(x))

prognozowane.etykietki.ucz.new <- klasy.ucz.new[maks.ind.ucz.new]
prognozowane.etykietki.test.new <- klasy.test.new[maks.ind.test.new]

rzeczywiste.etykietki.ucz.new <- etykietki.klas.ucz.new
rzeczywiste.etykietki.test.new <- etykietki.klas.test.new

# macierze pomyłek
macierz.pomylek.ucz.new <- table(rzeczywiste.etykietki.ucz.new, prognozowane.etykietki.u

macierz.pomylek.test.new <- table(rzeczywiste.etykietki.test.new, prognozowane.etykietki
```

```r
print(xtable(head(Y.ucz.hat.new), caption="Wartości prognozowane - nowy zbiór"))
```

	1	2	3
144	0.02	-0.17	1.15
130	-0.04	0.44	0.60
9	0.91	0.15	-0.06
83	0.05	0.93	0.02
105	0.02	-0.23	1.20
46	0.92	0.11	-0.03

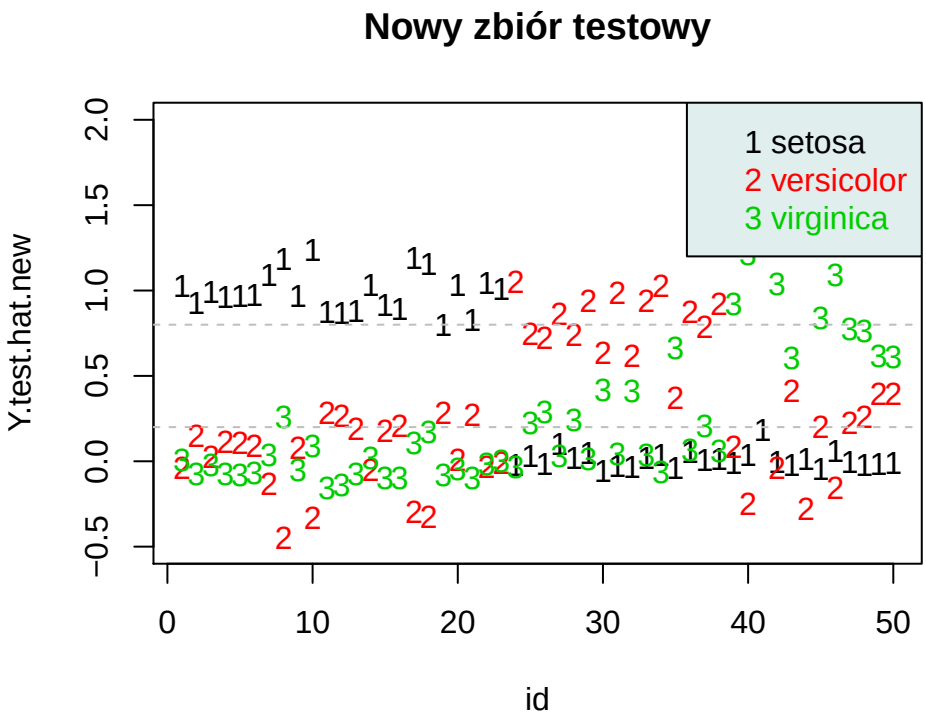
Tabela 6: Wartości prognozowane - nowy zbiór

```
par(mar = c(4, 4, 4, 1))
matplot(Y.ucz.hat.new, main="Nowy zbiór uczący",xlab="id", ylim=c(-.5,2))
abline(h=c(0.2,0.8), lty=2, col="gray")
legend(x="topright", legend=paste(1:3,levels(learning.set.new$Species)), col=1:3, text.c
```



Rysunek 5: Prognozy dla nowego zbioru uczącego

```
par(mar = c(4, 4, 4, 1))
matplot(Y.test.hat.new, main="Nowy zbiór testowy",xlab="id", ylim=c(-.5,2))
abline(h=c(0.2,0.8), lty=2, col="gray")
legend(x="topright", legend=paste(1:3,levels(test.set.new$Species)), col=1:3, text.col=1
```



Rysunek 6: Prognozy dla nowego zbioru testowego

```
print(xtable(macierz.pomylek.ucz.new, caption="Macierz pomyłek - nowy zbiór uczący"))
```

	setosa	versicolor	virginica
setosa	27	0	0
versicolor	0	35	0
virginica	0	2	36

Tabela 7: Macierz pomyłek - nowy zbiór uczący

```
# dokładność klasyfikacji - nowy zbiór uczący
sum(diag(macierz.pomylek.ucz.new))/n.ucz.new

## [1] 0.98
```

```
print(xtable(macierz.pomylek.test.new, caption="Macierz pomyłek - nowy zbiór testowy"))
```

```
# dokładność klasyfikacji - nowy zbiór testowy
sum(diag(macierz.pomylek.test.new))/n.test.new

## [1] 0.98
```

Wnioski:

	setosa	versicolor	virginica
setosa	23	0	0
versicolor	0	14	1
virginica	0	0	12

Tabela 8: Macierz pomyłek - nowy zbiór testowy

- Dokładność klasyfikacji w nowym, uzupełnionym modelu jest wyraźnie wyższa niż w pierwotnym.
- W tym przypadku nie występuje już problem maskowania klas.

2 Porównanie metod klasyfikacji

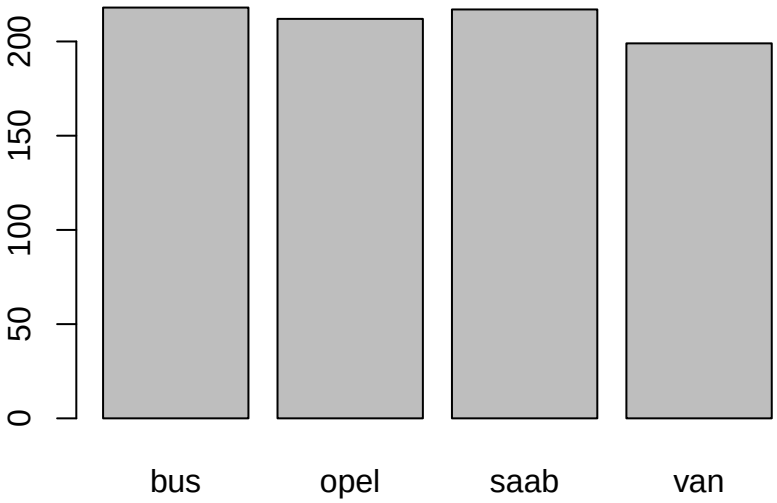
2.1 Wybór danych

Zadanie wykonamy z użyciem ramki danych **Vehicle** z biblioteki **mlbench**. Dane zawierają informacje na temat czterech typów pojazdów: dwupiętrowego autobusu, vana marki Chevrolet, Saaba 9000 oraz Opla Manty 400. Celem stworzenia tej ramki była klasyfikacja sylwetek aut za pomocą różnych obserwacji. W teorii rozróżnienie autobusu oraz vana powinno być łatwe, w przeciwieństwie do odróżnienia od siebie dwóch różnych modeli aut.

```
library("mlbench")
data("Vehicle")
attach(Vehicle)

#liczba kolumn
ncol(Vehicle)

## [1] 19

#liczba wierszy
nrow(Vehicle)

## [1] 846

#brakujące dane
sum(is.na(Vehicle))

## [1] 0
```

Jak widać w ramce nie brakuje żadnych wartości (nie są one również kodowane w niestandardowy sposób).

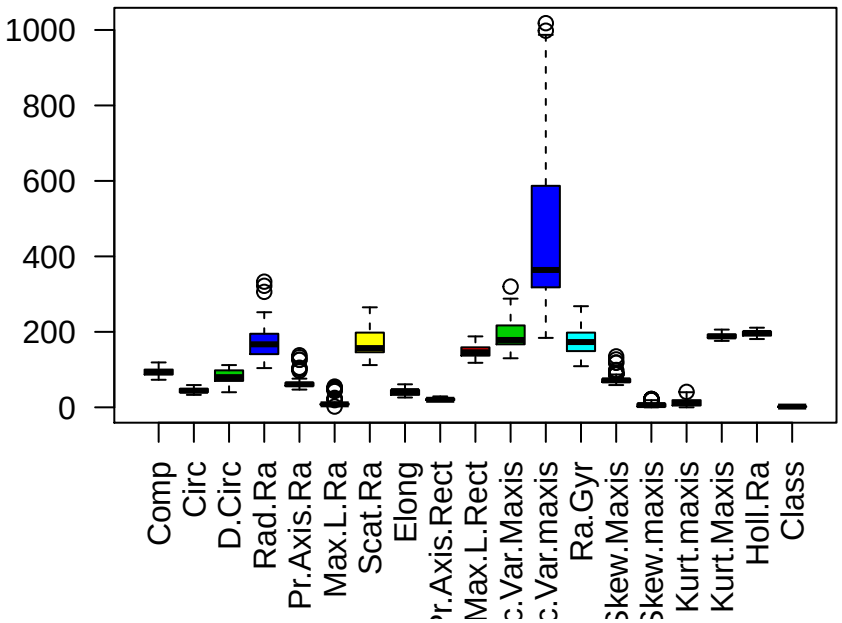
```
etykietki.veh <- Vehicle$Class
plot(etykietki.veh)
```



Rysunek 7: Przynależność do poszczególnych klas

Zbadajmy zmienność poszczególnych cech:

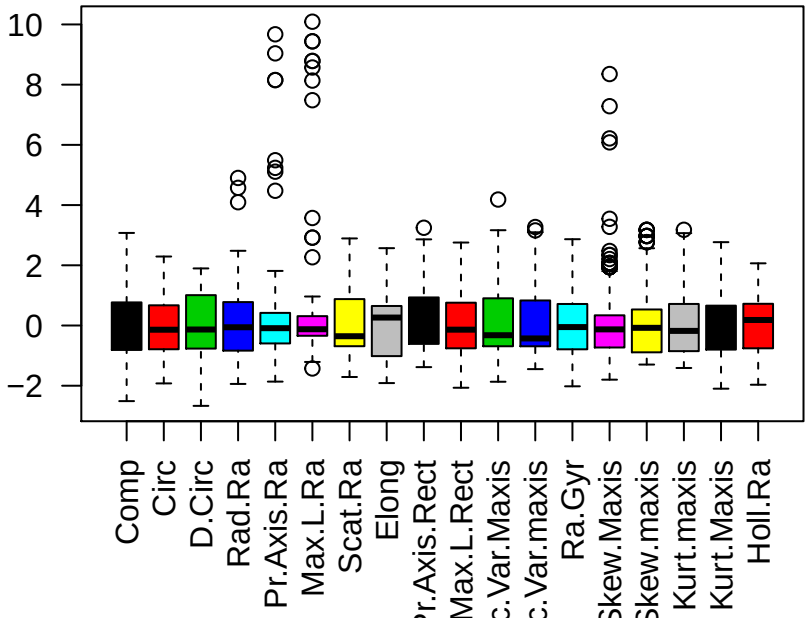
```
boxplot(Vehicle, las=2, col = 1:18)
```



Rysunek 8: Boxploty - badanie zmienności cech

Wariancje poszczególnych cech bardzo wyraźnie się od siebie różnią. Niezbędne będzie zatem zastosowanie standaryzacji.

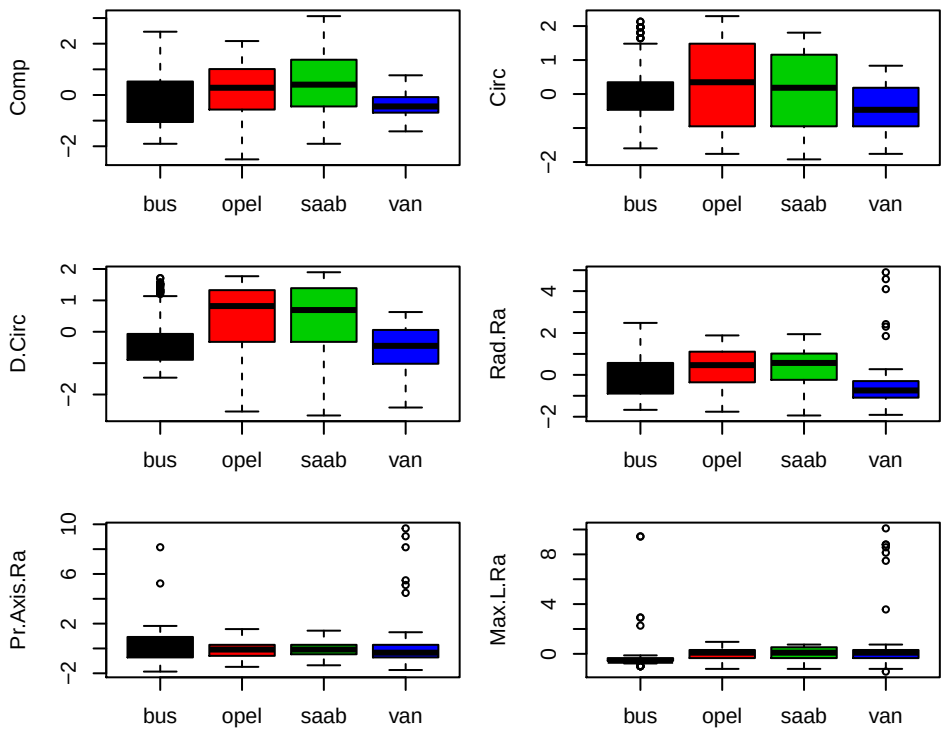
```
veh1 <- scale(Vehicle[1:18])
boxplot(veh1, las=2, col = 1:18)
```



Rysunek 9: Boxploty po standaryzacji

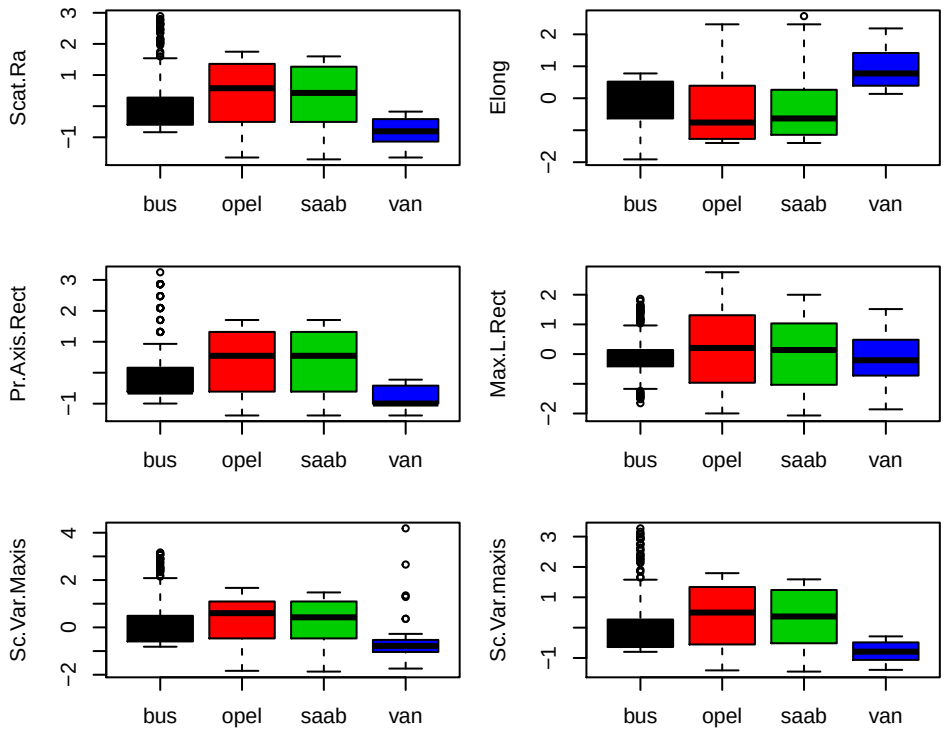
Postaramy się teraz wybrać cechy o najlepszej zdolności dyskryminacyjnej.

```
veh1 <- data.frame(veh1)
attach(veh1)
par(mfrow=c(3,2))
par(mar=c(3, 4, 1, 1))
boxplot(Comp~Vehicle$Class, col=1:4)
boxplot(Circ~Vehicle$Class, col=1:4)
boxplot(D.Circ~Vehicle$Class, col=1:4)
boxplot(Rad.Ra~Vehicle$Class, col=1:4)
boxplot(Pr.Axis.Ra~Vehicle$Class, col=1:4)
boxplot(Max.L.Ra~Vehicle$Class, col=1:4)
```



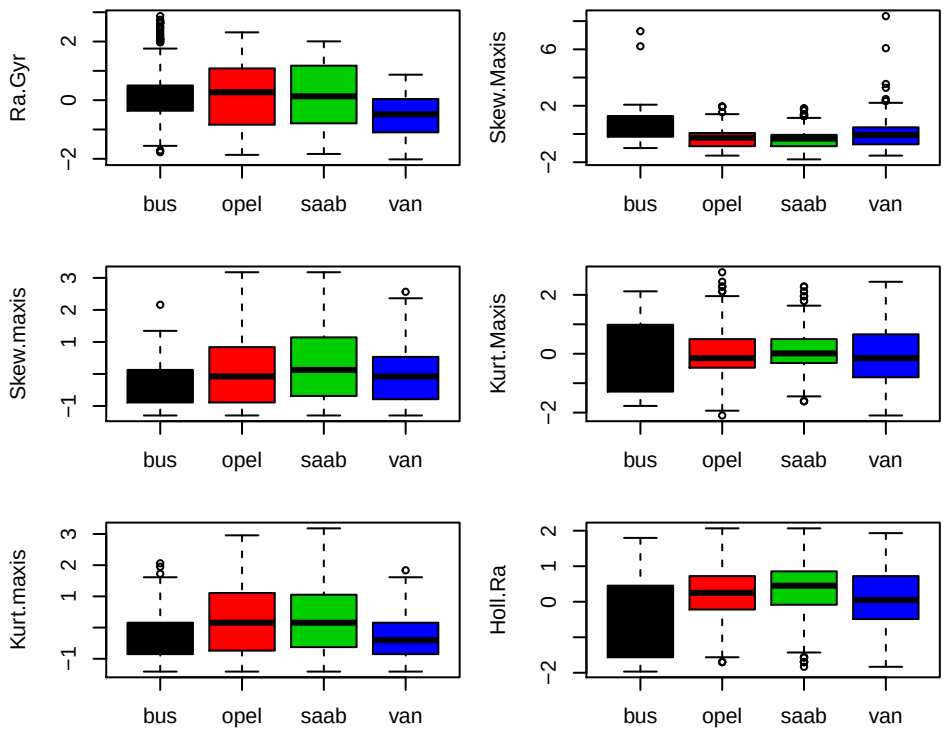
Rysunek 10: Zdolności dyskryminacyjne

```
par(mfrow=c(3,2))
par(mar=c(3, 4, 1, 1))
boxplot(Scat.Ra~Vehicle$Class, col=1:4)
boxplot(Elong~Vehicle$Class, col=1:4)
boxplot(Pr.Axis.Rect~Vehicle$Class, col=1:4)
boxplot(Max.L.Rect~Vehicle$Class, col=1:4)
boxplot(Sc.Var.Maxis~Vehicle$Class, col=1:4)
boxplot(Sc.Var.maxis~Vehicle$Class, col=1:4)
```



Rysunek 11: Zdolności dyskryminacyjne

```
par(mfrow=c(3,2))
par(mar=c(3, 4, 1, 1))
boxplot(Ra.Gyr~Vehicle$Class, col=1:4)
boxplot(Skew.Maxis~Vehicle$Class, col=1:4)
boxplot(Skew.maxis~Vehicle$Class, col=1:4)
boxplot(Kurt.Maxis~Vehicle$Class, col=1:4)
boxplot(Kurt.maxis~Vehicle$Class, col=1:4)
boxplot(Holl.Ra~Vehicle$Class, col=1:4)
```



Rysunek 12: Zdolności dyskryminacyjne

Wybór zmiennej o największej zdolności dyskryminacyjnej nie jest w tym przypadku jednoznaczny. Zdecydowaliśmy, że "najbardziej obiecująca" będzie kombinacja zmiennych Elong oraz Holl.Ra (w pierwszej wyróżnia się van, a w drugiej bus). Już na tym etapie zauważamy, że różnice między Oplem a Saabem rzeczywiście nie są wyraźne.

2.2 Podział danych na zbiór uczący i testowy

Podobnie jak w zadaniu pierwszym tworzymy dwa zbiory. Ponadto robimy to samo dla wcześniej wyselekcjonowanych przez nas zmiennych.

```
etykietki.veh <- data.frame(etykietki.veh)
veh2 <- cbind(veh1, etykietki.veh)

n.veh <- dim(veh2)[1]
learning.indx.veh <- sample(1:n.veh,2/3*n.veh)
learning.set.veh <- veh2[learning.indx.veh,]
test.set.veh <- veh2[-learning.indx.veh,]

veh3 <- cbind(veh1$Elong, veh1$Holl.Ra, etykietki.veh)

etykietki.learning.veh <- learning.set.veh$etykietki.veh
etykietki.test.veh <- test.set.veh$etykietki.veh

n.wybrane <- dim(veh3)[1]
learning.indx.wybrane <- sample(1:n.wybrane,2/3*n.wybrane)
```

```
learning.set.wybrane <- veh3[learning.indx.wybrane,]
test.set.wybrane <- veh3[-learning.indx.wybrane,]

etykietki.learning.wybrane <- learning.set.wybrane$etykietki.veh
etykietki.test.wybrane <- test.set.wybrane$etykietki.veh
```

2.3 Metoda k-najbliższych sąsiadów

```
library(ipred)

model.knn.veh <- ipredknn(etykietki.veh~ ., data=learning.set.veh, k=5)

etykietki.prog.learning.veh <- predict(model.knn.veh, learning.set.veh, type="class")
etykietki.prog.test.veh <- predict(model.knn.veh, test.set.veh, type="class")

# błąd klasyfikacji - funkcja
blad.klasyf <- function(etykietki.prog, etykietki)
{
  n <- length(etykietki.prog)
  macierz.pomylek <- table(etykietki.prog,etykietki)
  list(macierz.pomylek=macierz.pomylek, blad.klasyf=(n - sum(diag(macierz.pomylek))) / n
}
```

```
#zbiór uczący - wszystkie cechy
blad.klasyf(etykietki.prog.learning.veh, etykietki.learning.veh)

## $macierz.pomylek
##                 etykietki
## etykietki.prog bus opel saab van
##           bus  140    0    4   3
##           opel   1   85   26   3
##           saab   0   46  113   2
##           van    2   11    7 121
##
## $blad.klasyf
## [1] 0.1861702

#zbiór testowy - wszystkie cechy
blad.klasyf(etykietki.prog.test.veh, etykietki.test.veh)

## $macierz.pomylek
##                 etykietki
## etykietki.prog bus opel saab van
##           bus   71    2    1   4
##           opel   1   31   26   2
##           saab   0   32   34   0
##           van    3    5    6  64
##
## $blad.klasyf
## [1] 0.2907801
```

Jak widać metoda k-najbliższych sąsiadów daj niezłe rezultaty. Zobaczmy jakie efekty da metoda cross-validation:

```r
my.predict  <- function(model, newdata) predict(model, newdata=newdata, type="class")
my.ipredknn <- function(formula1, data1, ile.sasiadow) ipredknn(formula=formula1,data=da

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="cv", est.para=control.errorest(k = 10), ile.sasiadow=5)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "cv",
##     est.para = control.errorest(k = 10), ile.sasiadow = 5)
##
##   10-fold cross-validation estimator of misclassification error
##
## Misclassification error:  0.2754

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="boot", est.para=control.errorest(nboot = 50), ile.sasiadow=5)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "boot",
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 5)
##
##   Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.3095
## Standard deviation: 0.0027

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="632plus", est.para=control.errorest(nboot = 50), ile.sasiadow=5)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "632plus",
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 5)
##
##   .632+ Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.2704
```

Przetestujmy teraz tę metodę dla różnej liczby sąsiadów:

```r
errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="632plus", est.para=control.errorest(nboot = 50), ile.sasiadow=1)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "632plus",
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 1)
##
##   .632+ Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.2309

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="632plus", est.para=control.errorest(nboot = 50), ile.sasiadow=10)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "632plus",
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 10)
##
##   .632+ Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.2748

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="632plus", est.para=control.errorest(nboot = 50), ile.sasiadow=15)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "632plus",
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 15)
##
##   .632+ Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.2907

errorest(etykietki.veh ~., veh2, model=my.ipredknn, predict=my.predict,
         estimator="632plus", est.para=control.errorest(nboot = 50), ile.sasiadow=20)

##
## Call:
## errorest.data.frame(formula = etykietki.veh ~ ., data = veh2,
##     model = my.ipredknn, predict = my.predict, estimator = "632plus",
```

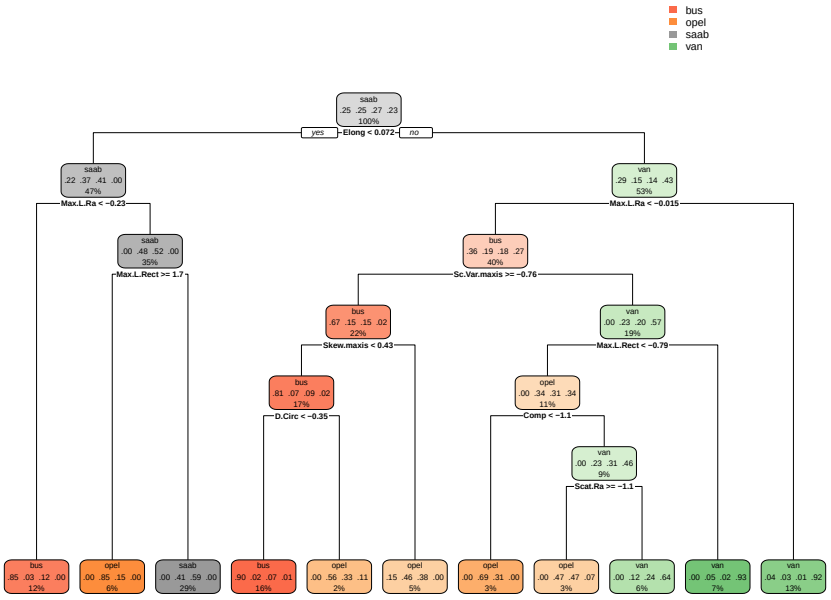
```
##     est.para = control.errorest(nboot = 50), ile.sasiadow = 20)
##
##   .632+ Bootstrap estimator of misclassification error
##   with 50 bootstrap replications
##
## Misclassification error:  0.3033
```

Jak widać w naszym przypadku wraz ze wzrostem liczby sąsiadów rośnie też błąd klasyfikacyjny.

2.4 Drzewa klasyfikacyjne

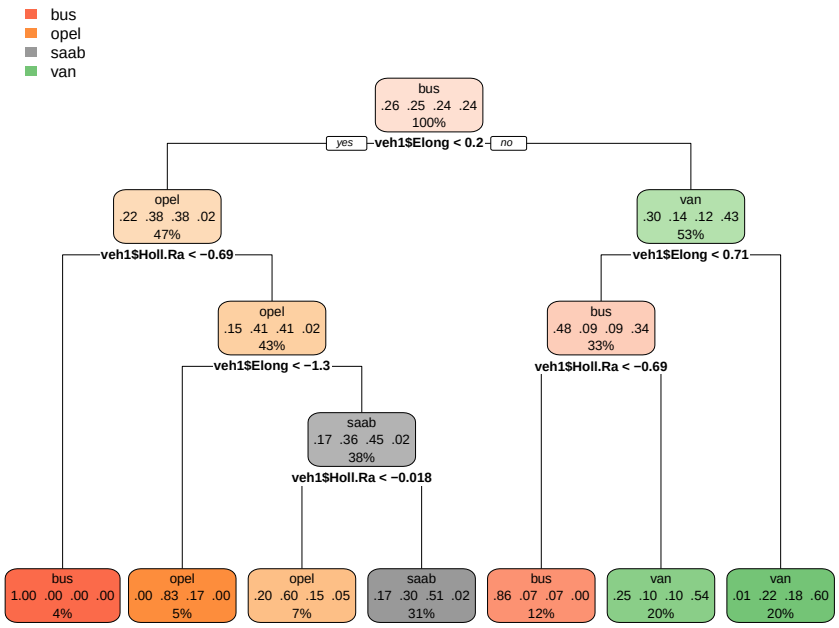
```
library(rpart)
library(rpart.plot)
model <- etykietki.veh~ .
veh.tree.simple <- rpart(model, data=learning.set.veh)
veh.tree.simple.wybrane <-rpart(model, data=learning.set.wybrane)

# prognozy dla zbioru uczącego
pred.labels.learning <- predict(veh.tree.simple, newdata=learning.set.veh, type = "class
pred.labels.learning.wybrane <- predict(veh.tree.simple.wybrane, newdata=learning.set.wy

# prognozy dla zbioru testowego
pred.labels.test <- predict(veh.tree.simple, newdata=test.set.veh, type = "class")
pred.labels.test.wybrane <- predict(veh.tree.simple.wybrane, newdata=test.set.wybrane, t

# wyznaczenie prognozowanych prawdopodobieństw a posteriori
pred.probs.test <- predict(veh.tree.simple, newdata=test.set.veh, type = "prob")
pred.probs.test.wybrane <- predict(veh.tree.simple.wybrane, newdata=test.set.wybrane, ty
```

```
rpart.plot(veh.tree.simple)
```



Rysunek 13: Drzewo klasyfikacyjne - wszystkie cechy

```
rpart.plot(veh.tree.simple.wybrane)
```



Rysunek 14: Drzewo klasyfikacyjne - wybrane cechy

```
# zbiór uczący - wszystkie
conf.mat.learning <- table(pred.labels.learning, learning.set.veh$etykietki.veh)
conf.mat.learning

##
## pred.labels.learning bus opel saab van
##                 bus  136    4   14   1
##                 opel   4   63   30   2
##                 saab   0   67   96   0
##                 van    3    8   10 126

(error.rate.learning <- (nrow(learning.set.veh) - sum(diag(conf.mat.learning))) / nrow(l

## [1] 0.2535461

#zbiór testowy - wszystkie
conf.mat.test <- table(pred.labels.test, test.set.veh$etykietki.veh)
conf.mat.test

##
## pred.labels.test bus opel saab van
##             bus   63    7    7   1
```

```
##             opel   8   25   20   9
##             saab   1   34   32   0
##             van    3    4    8  60

(error.rate.test <- (nrow(test.set.veh) - sum(diag(conf.mat.test))) / nrow(test.set.veh)

## [1] 0.3617021

# zbiór uczący - wybrane
conf.mat.learning.wybrane <- table(pred.labels.learning.wybrane, learning.set.wybrane$et
conf.mat.learning.wybrane

##
## pred.labels.learning.wybrane bus opel saab van
##                         bus   81    5    5   0
##                         opel   8   48   11   2
##                         saab  29   53   90   3
##                         van   30   37   32 130

(error.rate.learning.wybrane <- (nrow(learning.set.wybrane) - sum(diag(conf.mat.learning

## [1] 0.3812057

# zbiór testowy - wybrane
conf.mat.test.wybrane <- table(pred.labels.test.wybrane, test.set.wybrane$etykietki.veh)
conf.mat.test.wybrane

##
## pred.labels.test.wybrane bus opel saab van
##                      bus   34    3    5   0
##                      opel   6   17   11   1
##                      saab  18   34   44   0
##                      van   12   15   19  63

(error.rate.test.wybrane <- (nrow(test.set.wybrane) - sum(diag(conf.mat.test.wybrane)))

## [1] 0.4397163
```

W przypadku drzew klasyfikacynych zbiory złożone ze wszystkich cech sprawdzają się lepiej od tych zawierające jedynie wybrane wcześniej cechy.

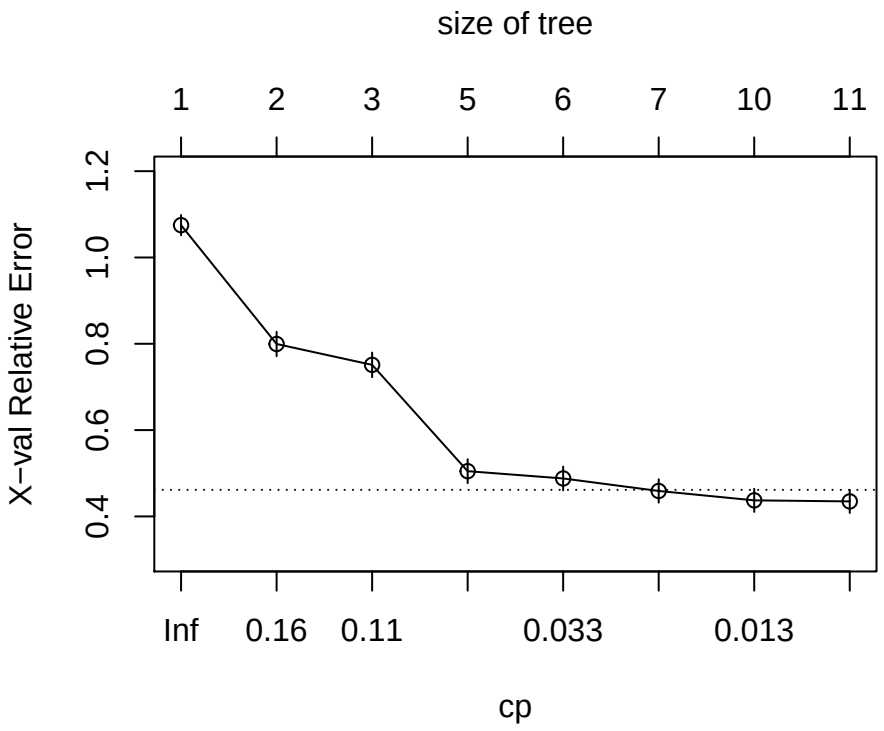
```
# Zmiana parametrów -- konstruujemy złożone drzewo
veh.tree.complex  <- rpart(model, data=learning.set.veh, control=rpart.control(cp=.01, m

# Wybór parametru złożoności (cp)
printcp(veh.tree.complex)

##
## Classification tree:
```

```
## rpart(formula = model, data = learning.set.veh, control = rpart.control(cp = 0.01,
##     minsplit = 5, maxdepth = 20))
##
## Variables actually used in tree construction:
## [1] Comp         D.Circ       Elong        Max.L.Ra     Max.L.Rect
## [6] Sc.Var.maxis Scat.Ra      Skew.maxis
##
## Root node error: 414/564 = 0.73404
##
## n= 564
##
##         CP nsplit rel error  xerror     xstd
## 1 0.212560      0   1.00000 1.07488 0.023405
## 2 0.118357      1   0.78744 0.79952 0.028246
## 3 0.097826      2   0.66908 0.75121 0.028530
## 4 0.055556      4   0.47343 0.50483 0.027704
## 5 0.019324      5   0.41787 0.48792 0.027504
## 6 0.013285      6   0.39855 0.45894 0.027113
## 7 0.012077      9   0.35749 0.43720 0.026779
## 8 0.010000     10   0.34541 0.43478 0.026740
```

```
bestcp <- veh.tree.complex$cptable[which.min(veh.tree.complex$cptable[,"xerror"]),"CP"]
bestcp # najmniejszy błąd
```

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

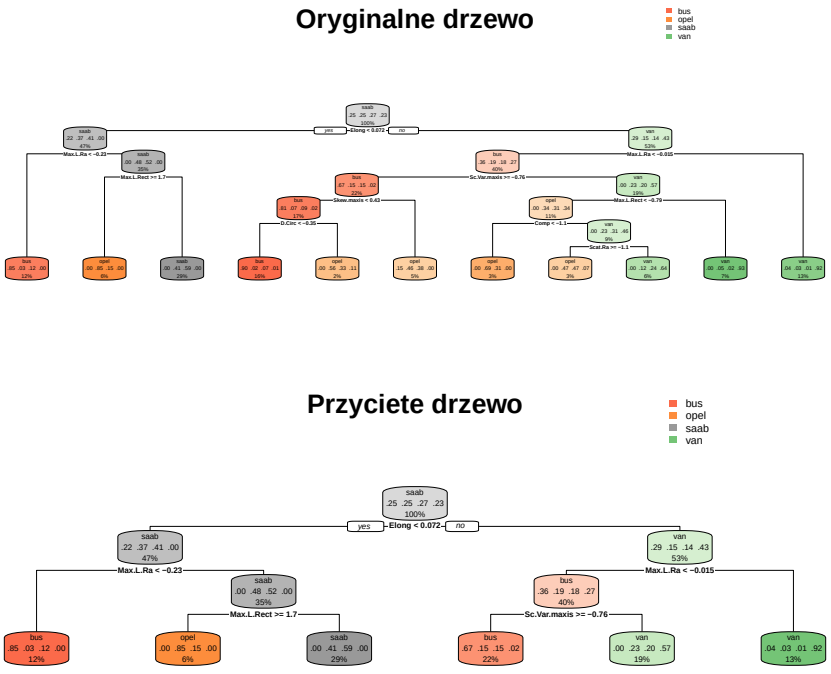
```
plotcp(veh.tree.complex)
```



Rysunek 15: Wybór parametru złożoności

```
cp.opt <- 0.02

# Przycinanie drzew na podstawie kryterium kosztu-złożoności
veh.tree.complex.pruned <- prune(veh.tree.complex, cp = cp.opt)
```

```
par(mfrow=c(2,1))
rpart.plot(veh.tree.complex, main="Oryginalne drzewo")
rpart.plot(veh.tree.complex.pruned, main="Przyciete drzewo")
```



Rysunek 16: Drzewo oryginalne i przycięte

2.5 Naiwny klasyfikator Bayesowski

```
library(MASS)
library(klaR)
library("e1071")

model.NB <- naiveBayes(learning.set.veh$etykietki.veh~., data = learning.set.veh)

model.NB.wybrane <- naiveBayes(learning.set.wybrane$etykietki.veh~., data=learning.set.w

etykietki.prog.learning.veh <- predict(model.NB, learning.set.veh)
etykietki.prog.learning.wybrane <- predict(model.NB.wybrane, learning.set.wybrane)

etykietki.prog.test.veh <- predict(model.NB, test.set.veh)
etykietki.prog.test.wybrane <- predict(model.NB.wybrane, test.set.wybrane)
```

```
#zbiór uczący - wszystkie cechy
blad.klasyf(etykietki.prog.learning.veh, etykietki.learning.veh)

## $macierz.pomylek
##                etykietki
## etykietki.prog bus opel saab van
##           bus   26    2    3   8
##           opel  13   37   10   0
##           saab  17   61   99   6
##           van   87   42   38 115
##
## $blad.klasyf
## [1] 0.5088652

#zbiór testowy - wszystkie cechy
blad.klasyf(etykietki.prog.test.veh, etykietki.test.veh)

## $macierz.pomylek
##                etykietki
## etykietki.prog bus opel saab van
##           bus   15    0    1   5
##           opel   9   13    3   1
##           saab   9   36   36   3
##           van   42   21   27  61
##
## $blad.klasyf
## [1] 0.5567376

#zbiór uczący - wybrane cechy
blad.klasyf(etykietki.prog.learning.wybrane, etykietki.learning.wybrane)
```

```
## $macierz.pomylek
##                etykietki
## etykietki.prog bus opel saab van
##           bus   84   11   11  16
##           opel  15   58   42  10
##           saab  27   38   60   7
##           van   22   36   25 102
##
## $blad.klasyf
## [1] 0.4609929

#zbiór testowy - wybrane cechy
blad.klasyf(etykietki.prog.test.wybrane, etykietki.test.wybrane)

## $macierz.pomylek
##                etykietki
## etykietki.prog bus opel saab van
##           bus   37    4    4  11
##           opel   6   28   28   4
##           saab  19   23   27   6
##           van    8   14   20  43
##
## $blad.klasyf
## [1] 0.5212766
```

Naiwny klasyfikator Bayesowski w naszym przypadku sprawdza się słabo - błąd klasyfikacyjny oscylujący wokół 50% ciężko uznać choćby za zadowalający. Zbiory z wybranymi cechami dają zbliżone rezultaty do tych ze wszystkimi.

2.6 Wnioski

- W naszym przypadku najlepsza okazała się metoda k-najbliższych sąsiadów.
- Zadowalające wyniki udało się uzyskać również dzięki metodzie drzew klasyfikacyjnych.
- Zdecydowanie najsłabiej wypadł naiwny klasyfikator Bayesa.