

# Raport 3

## Eksploracja danych

Mikołaj Langner, Marcin Kostrzewa  
nr albumów: 255716, 255749

2021-04-19

## Spis treści

<b>1</b>	<b>Wstęp</b>	<b>1</b>
<b>2</b>	<b>Zadanie 1</b>	<b>2</b>
2.1	Wczytanie danych i podział na zbiór uczący i testowy . . . . .	2
2.2	Konstrukcja klasyfikatora i wyznaczenie prognoz . . . . .	2
2.3	Ocena jakości klasyfikacji . . . . .	4
2.4	Zastosowanie regresji liniowej do modelu o rozszerzonej ilości cech . . . . .	4
<b>3</b>	<b>Zadanie 2</b>	<b>7</b>
3.1	Wczytanie i krótka analiza danych . . . . .	7
3.2	Metoda k-najbliższych sąsiadów . . . . .	8
3.3	Drzewa klasyfikacyjne . . . . .	8
3.4	Naiwny klasyfikator bayesowski . . . . .	9

## 1 Wstęp

Raport zawiera rozwiązania listy 3.

W zadaniu pierwszym budujemy klasyfikator na bazie metody regresji liniowej i oceniamy jego skuteczność i dokładność.

W zadaniu drugim . . . Porównamy ze sobą rezultaty zastosowania:

- metoda k-najbliższych sąsiadów (*k-Nearest Neighbors*),
- drzewa klasyfikacyjne (*classification trees*),
- naiwny klasyfikator bayesowski (*naïve Bayes classifier*).

## 2 Zadanie 1

### 2.1 Wczytanie danych i podział na zbiór uczący i testowy

Wczytajmy dane o irysach i podzielmy je na zbiór uczący i testowy w proporcji 1 : 2.

```
data(iris)
n <- dim(iris)[1]

train.set.index <- sample(1:n, 2/3*n)
train.set <- iris %>% slice(train.set.index) %>% arrange(Species)
test.set <- iris %>% slice(-train.set.index) %>% arrange(Species)
```

### 2.2 Konstrukcja klasyfikatora i wyznaczenie prognoz

Stworzmy teraz macierze eksperymentu i wskaźnikową zarówno dla zbioru uczącego, jak i testowego. W tym celu wykorzystamy funkcję `dummyVars` z pakietu `Caret`.

```
dummies <- dummyVars(" ~ .", data=iris)

train.dummies <- predict(dummies, newdata = train.set)
train.X <- as.matrix(cbind(rep(1, nrow(train.dummies)),
                           train.dummies[, 1:4]))
train.Y <- train.dummies[, 5:7]

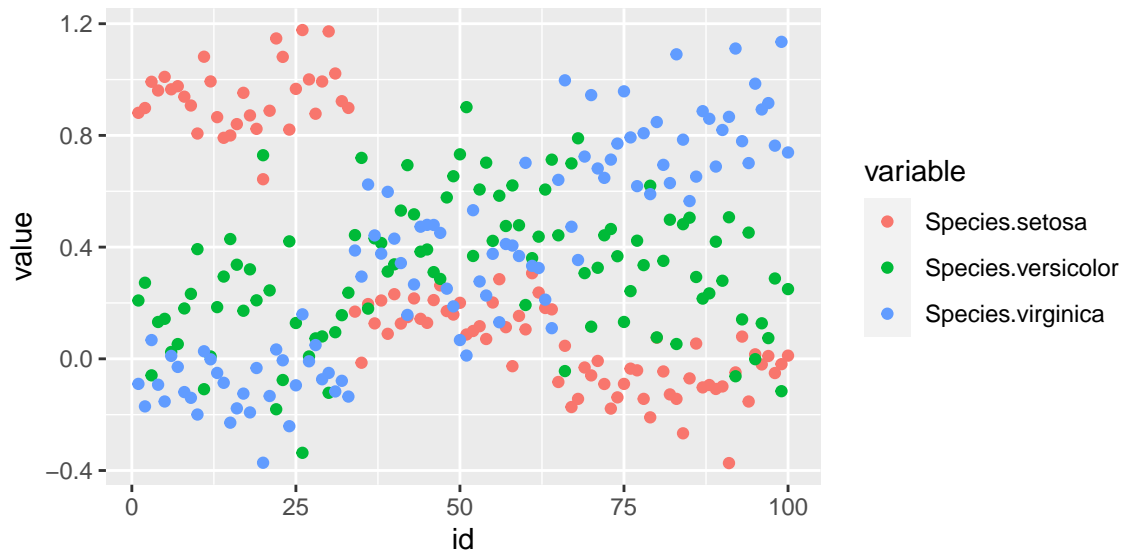
test.dummies <- predict(dummies, newdata = test.set)
test.X <- as.matrix(cbind(rep(1, nrow(test.dummies)), test.dummies[, 1:4]))
test.Y <- test.dummies[, 5:7]
```

Wykorzystując metodę najmniejszych kwadratów, wyznaczamy przewidywane prognozy klas dla obu zbiorów.

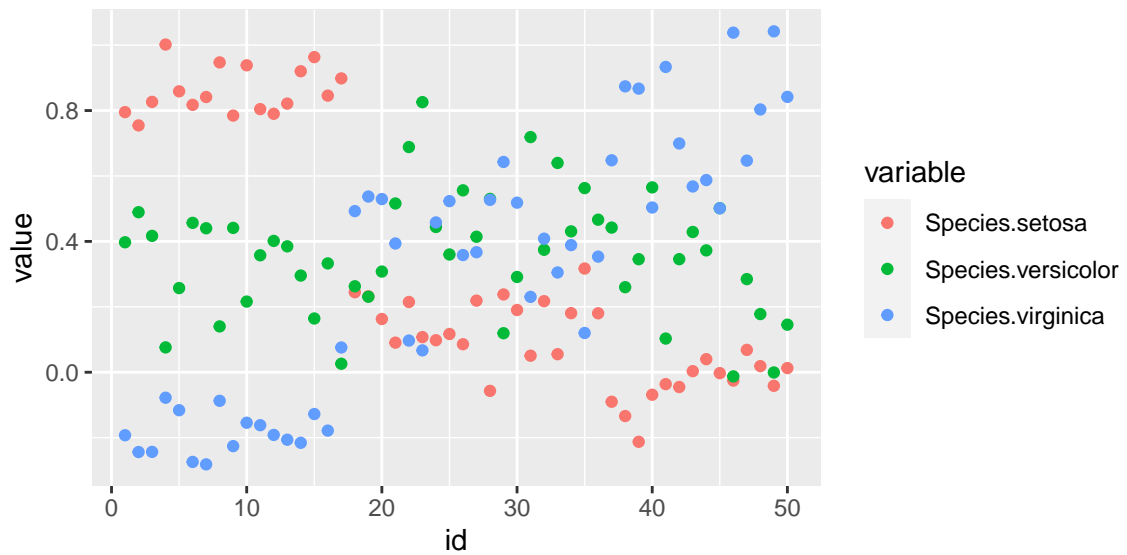
```
Y.hat <- solve(t(train.X) %*% train.X) %*% t(train.X) %*% train.Y

train.proba <- train.X %*% Y.hat
test.proba <- test.X %*% Y.hat
```

Przedstawmy prognozy klas na wykresach.



Rysunek 1: Prognozy klas dla zbioru uczącego.



Rysunek 2: Prognozy klas dla zbioru testowego.

## 2.3 Ocena jakości klasyfikacji

Wyznaczymy teraz macierz pomyłek dla zbioru uczącego.

Tabela 1: Macierz pomyłek dla zbioru uczącego.

	Species.setosa	Species.versicolor	Species.virginica
setosa	32	1	0
versicolor	0	21	10
virginica	0	3	33

Błąd klasyfikacji to 0.14.

Tabela 2: Macierz pomyłek dla zbioru testowego.

	Species.setosa	Species.versicolor	Species.virginica
setosa	17	0	0
versicolor	0	11	8
virginica	0	2	12

Błąd klasyfikacji wynosi 0.2.

Wnioski i napomnienie o maskowaniu

## 2.4 Zastosowanie regresji liniowej do modelu o rozszerzonej ilości cech

Najpierw uzupełnimy dane o irysach o składniki wielomianowe stopnia 2.

```
iris.quad <- (iris %>% select(-Species))^2
colnames(iris.quad) <- c("SL^2", "SW^2", "PL^2", "PW^2")
iris <- cbind(iris, combn(iris %>% select(-Species), 2,
                        FUN = Reduce, f = `*`),
             iris.quad)
```

Podobnie jak poprzednio podzielimy dane na zbiory: uczący i testowy, a następnie utworzymy macierze: eksperymentu i indykatorów.

```
train.set.index <- sample(1:n, 2/3*n)
train.set <- iris %>% slice(train.set.index) %>% arrange(Species)
test.set <- iris %>% slice(-train.set.index) %>% arrange(Species)

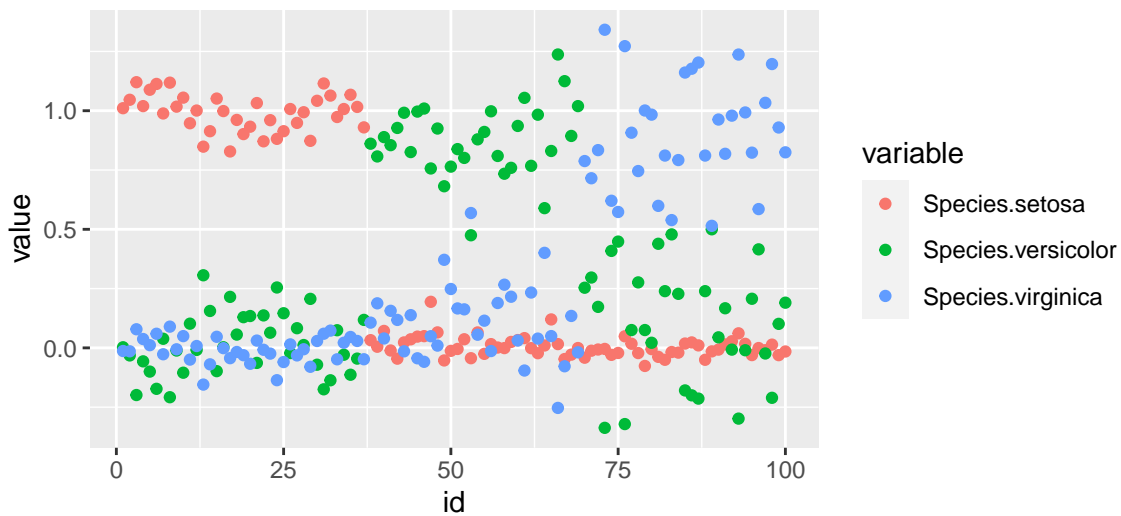
dummies <- dummyVars(" ~ .", data=iris)
train.dummies <- predict(dummies, newdata = train.set)
train.X <- as.matrix(cbind(rep(1, nrow(train.dummies)), train.dummies[, -c(5:7)]))
train.Y <- train.dummies[, 5:7]
test.dummies <- predict(dummies, newdata = test.set)
```

```
test.X <- as.matrix(cbind(rep(1, nrow(test.dummies)), test.dummies[, -c(5:7)]))
test.Y <- test.dummies[, 5:7]
```

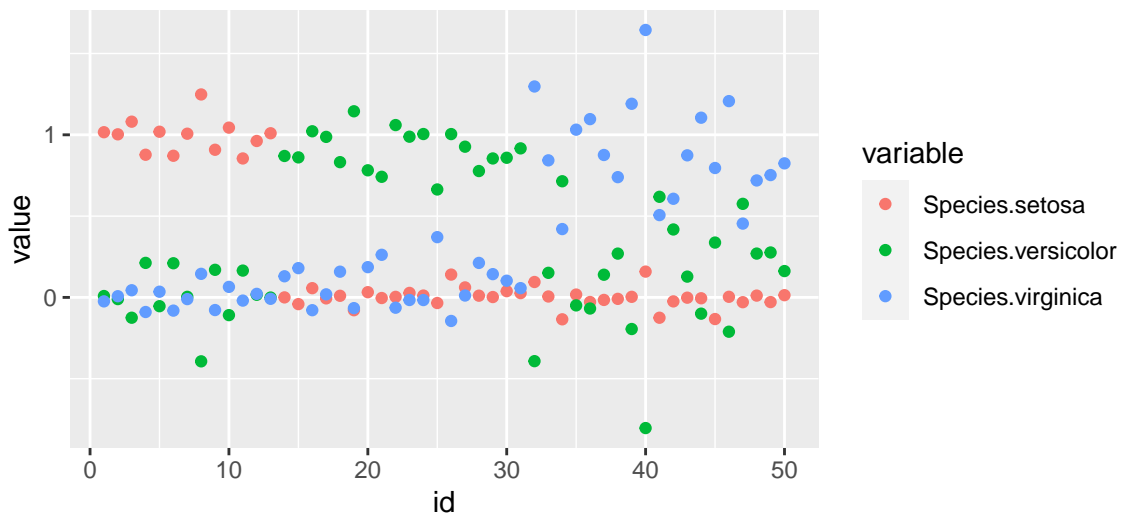
Ponownie, wyznaczmy prognozy klas i zwizualizujemy to przypisanie na wykresach.

```
Y.hat <- solve(t(train.X) %*% train.X) %*% t(train.X) %*% train.Y

train.proba <- train.X %*% Y.hat
test.proba <- test.X %*% Y.hat
```



Rysunek 3: Prognozy klas dla zbioru uczacego o rozszerzonej liczbie cech.



Rysunek 4: Prognozy klas dla zbioru uczacego o rozszerzonej liczbie cech.

Wyznaczymy także macierze pomyłek i błędy klasyfikacji.

Tabela 3: Macierz pomyłek dla zbioru uczącego dla przypadku o rozszerzonej liczbie cech.

	Species.setosa	Species.versicolor	Species.virginica
setosa	37	0	0
versicolor	0	31	1
virginica	0	0	31

Tabela 4: Macierz pomyłek dla zbioru testowego dla przypadku o rozszerzonej liczbie cech.

	Species.setosa	Species.versicolor	Species.virginica
setosa	13	0	0
versicolor	0	18	0
virginica	0	3	16

Błąd klasyfikacji wynosi 0.01.

Błąd klasyfikacji wynosi 0.06.

Wnioski i napomnienie o maskowaniu

## 3 Zadanie 2

### 3.1 Wczytanie i krótka analiza danych

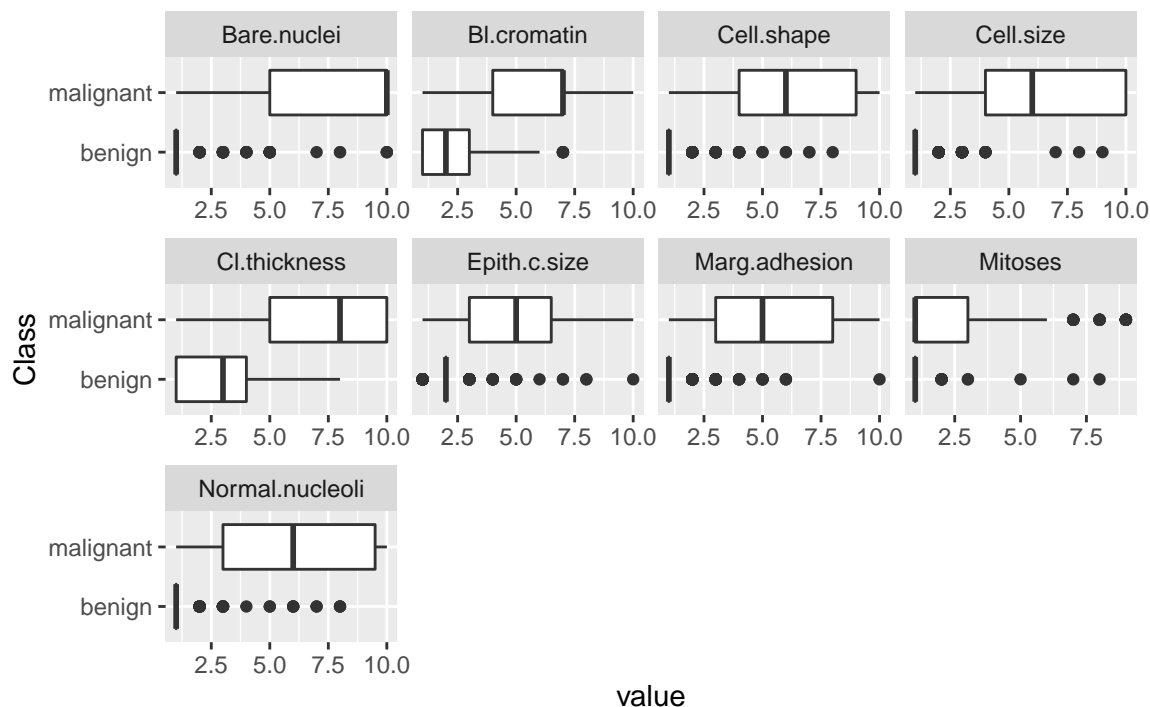
Wczytajmy i przygotujmy dane do dalszych analizy.

```
data("BreastCancer")
n <- dim(BreastCancer)[1]

BreastCancer <- BreastCancer %>% select(-Id)
BreastCancer <- drop_na(BreastCancer)
BreastCancer[, 1:9] <- data.frame(sapply(BreastCancer[, 1:9], as.numeric))

for (column in colnames(BreastCancer)) {
  if (is.factor(BreastCancer[, column]) & column != "Class") {
    BreastCancer[, column] <- ordered(BreastCancer[, column])
  }
}
```

Przyjrzyjmy się naszym danym na wykresach pudełkowych.



Możemy zauważyć, że zmiennymi, które dobrze ...

Podzielmy nasze dane na zbiór uczący i testowy.

```

set.seed(42)
train.index <- sample(n, n/7)
train.data <- BreastCancer %>% slice(train.index)
test.data <- BreastCancer %>% slice(-train.index)

train.subset <- train.data[, c(3, 6, 10)]
test.subset <- test.data[, c(3, 6, 10)]

train.etiquettes <- train.data$Class
test.etiquettes <- test.data$Class
subset.train.etiquettes <- train.subset$Class
subset.test.etiquettes <- test.subset$Class

cv <- trainControl(method="cv", number=5)

```

## 3.2 Metoda k-najbliższych sąsiadów

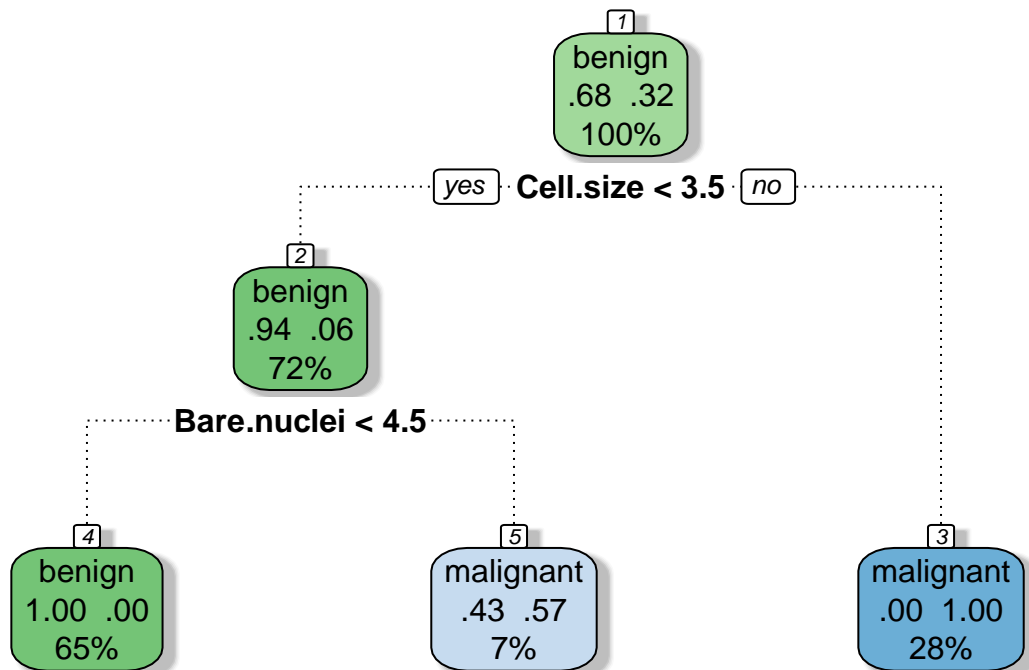
Coś tam, coś tam

## 3.3 Drzewa klasyfikacyjne

```

basic.tree.model <- rpart(Class ~ ., data = train.data)
fancyRpartPlot(basic.tree.model, sub="")

```





### 3.4 Naiwny klasyfikator bayesowski

```

bayes.model.basic <- naiveBayes(Class ~ ., data = train.data)

basic.bayes.train.pred <- predict(bayes.model.basic, train.data)
basic.bayes.test.pred <- predict(bayes.model.basic, test.data)

test.confusion <- table(test.etiquettes, basic.bayes.test.pred)
train.confusion <- table(train.etiquettes, basic.bayes.train.pred)
test.matrix <- as.matrix(test.confusion)
train.matrix <- as.matrix(train.confusion)

test.confusion %>% kbl(caption="Macierz pomylek dla zbioru testowego.", format="latex",

```

Tabela 5: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	357	23
malignant	2	207

```

test.matrix %>% kbl(caption="Macierz pomylek dla zbioru testowego.", format="latex", dig

```

Tabela 6: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	357	23
malignant	2	207

```

print(xtable(train.matrix), file="ta.tex", floating=FALSE)
print(xtable(test.matrix), file="tb.tex", floating=FALSE)

```

	benign	malignant
benign	62	2
malignant	0	30

(a) Table x(a)

	benign	malignant
benign	357	23
malignant	2	207

(b) Table x(b)

Tabela 7: Caption about here

Błędy klasyfikacji to kolejno 0.0212766 i 0.0424448.

Powtórzmy teraz powyższe dla wybranego podzbioru naszych danych.

```

bayes.model.subset <- naiveBayes(Class ~ ., data = train.subset)

basic.bayes.train.pred <- predict(bayes.model.subset, train.subset)
basic.bayes.test.pred <- predict(bayes.model.subset, test.subset)

```

```
test.confusion <- table(subset.test.etiquettes, basic.bayes.test.pred)
train.confusion <- table(subset.train.etiquettes, basic.bayes.train.pred)
test.matrix <- as.matrix(test.confusion)
train.matrix <- as.matrix(train.confusion)

train.matrix %>% kbl(caption="Macierz pomylek dla zbioru testowego.", format="latex", digits=2)
```

Tabela 8: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	62	2
malignant	1	29

```
test.matrix %>% kbl(caption="Macierz pomylek dla zbioru testowego.", format="latex", digits=2)
```

Tabela 9: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	360	20
malignant	7	202

```
print(xtable(train.matrix), file="tc.tex", floating=FALSE)
print(xtable(test.matrix), file="td.tex", floating=FALSE)
```

Błędy klasyfikacji to kolejno 0.0319149 i 0.0458404.

Model “stunigowany”

```
model <- train(Class ~ ., data = train.data, method = "naive_bayes", trControl = cv)
test.confusion <- table(test.data$Class, predict(model, test.data))
train.confusion <- table(train.data$Class, predict(model, train.data))

train.confusion %>% as.matrix %>% kbl(caption="Macierz pomylek dla zbioru testowego.", digits=2)
```

Tabela 10: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	63	1
malignant	2	28

```
test.confusion %>% as.matrix %>% kbl(caption="Macierz pomylek dla zbioru testowego.", digits=2)
```

Błędy klasyfikacji w tym przypadku to kolejno 0.0319149 i 0.0288625.

```
model <- train(Class ~ ., data = train.data, method = "knn", trControl = cv)
confusion <- table(test.data$Class, predict(model, test.data))
confusion
```

Tabela 11: Macierz pomylek dla zbioru testowego.

	benign	malignant
benign	369	11
malignant	6	203

```
##
##           benign malignant
##  benign      369         11
##  malignant     9         200

sum(diag(confusion)) / nrow(test.data)

## [1] 0.9660441

model <- train(Class ~ ., data = train.data, method = "rpart", trControl = cv)
confusion <- table(test.data$Class, predict(model, test.data))
confusion

##
##           benign malignant
##  benign      369         11
##  malignant    33         176

sum(diag(confusion)) / nrow(test.data)

## [1] 0.9252971

# my.predict <- function(model, newdata)
# { predict(model, newdata=newdata) }
#
#
# cv <- trainControl(method="cv", number=5)
#
#
# my.naiveBayes <- function(formula, data)
# { train(formula, data = data, method = "naive_bayes", trControl = cv)}
#
#
# my.tree <- function(formula, data)
# { }
#
#
# my.knn <- function(formula, data)
# { }
#
#
```

```
# blad.cv <- errorest(Class~., BreastCancer, model=my.naiveBayes, predict=my.predict,  
#                       estimator="cv", est.para=control.errorest(k = 5))
```