SPRAWOZDANIE

Zajęcia: Analiza Procesów Uczenia
Prowadzący: prof. Dr hab. Vasyl Martsenyuk

Laboratorium 5
3.02.2021
Temat: „Modelowanie procesów uczenia maszynowego w pakiecie mlr. Trenowanie, ocena i porównywanie modeli w pakiecie mlr.”
Wariant 8

Kamil Pająk
Informatyka II stopień
Stacjonarne (zaoczne)
1 semestr

1. Polecenie:

Zadanie 1 dotyczy konstruowania drzew decyzyjnych oraz reguł klasyfikacyjnych na podstawie zbioru danych (library(MASS lub datasets)). Wariant zadania genotype

Zadanie 2 dotyczy prognozowania oceny klientów (w skali 5-punktowej, Error < 5%) urządzeń RTV AGD, określonych na Zajęciu 1. Rozwiązanie polega na użyciu pakietu mlr. Należy wybrać najlepszą metodę wśród 5 możliwych z punktu widzenia przecyzyjności. Wyniki porównywania precyzyjności metod należy przedstawić w postaci graficznej

2. Wprowadzane dane:

Cały program znajduję się na Githubie:
https://github.com/vincidaking/APU

```r
library(caret)
file_data <- read.csv("monitors.csv")

dataset <- file_data
train_data_precent <- 0.7
ind <- createDataPartition(dataset, p =
                             train_data_precent, list = F)
train_data <- dataset[ind,]
test_data<- dataset[-ind,]


dataset$Ocena <- as.factor(dataset$Ocena)

#create rpart tree
library(rpart)
library(rpart.plot)
train_data_rpart <- rpart(Ocena  ~Cena, data = train_data)
test_data_rpart <- rpart(Ocena ~Cena, data = test_data)
rpart.plot(train_data_rpart)
train_data_rpart


#create binary tree
#library(party)
#train_data_ctree <- ctree(cena ~ ., data = train_data)
#test_data_ctree <- ctree(cena ~ ., data = test_data)
#plot(train_data_ctree)

dataset$Ocena <- as.factor(dataset$Ocena)

library(mlr)
listLearners()
#create task for dataset
task <- makeClassifTask(id = 'dataset',
                        data = dataset, target = "Ocena");
lrn <- makeLearner("classif.rpart", predict.type = "prob")
rdesc = makeResampleDesc("Bootstrap", iters = 100)
r = resample(learner = lrn, task = task, resampling = rdesc, show.info = T)

library(C50)
library(rFerns)
library(randomForestSRC)
lrns <- makeLearners(c("rpart", "C50","rFerns","randomForestSRC"),
                     type = "classif")
benchmark_result <- benchmark(learners = lrns, tasks = task,
                              resampling = cv5)

plt = plotBMRBoxplots(benchmark_result, measure = mmce, order.lrn =
                        getBMRLearnerIds(benchmark_result))
levels(plt$data$task.id) = c("genotype")
levels(plt$data$learner.id) = c("rpart", "C50","rFerns","randomForestSRC")

plt + ylab("Error rate")
```

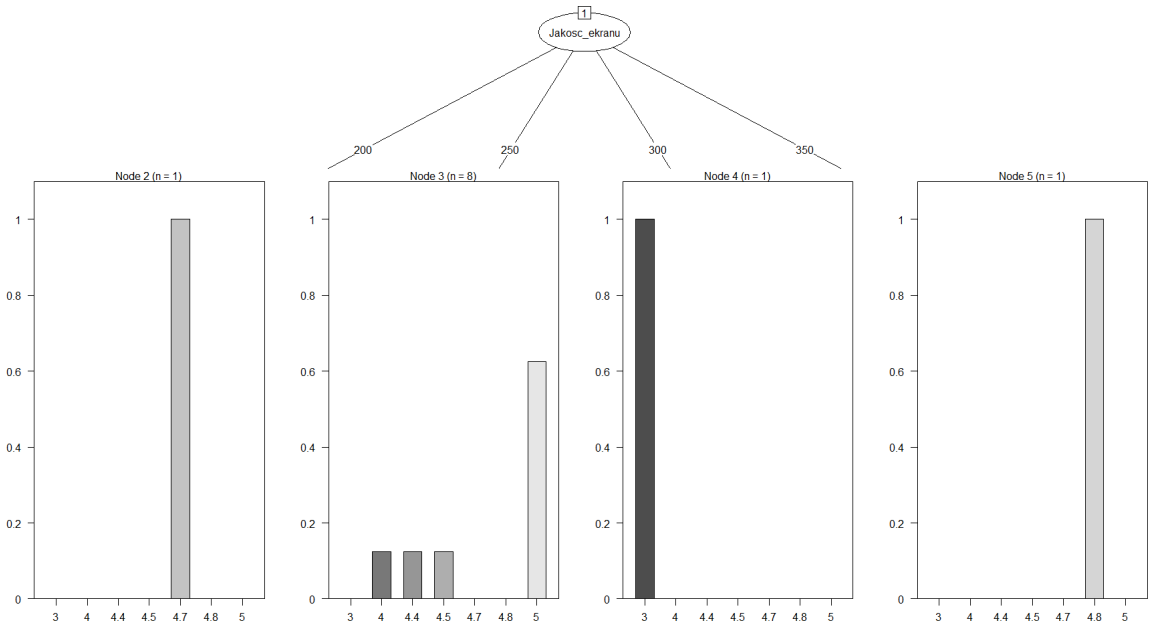
```r
library(MASS)


#split database for train i test
library(caret)
data(genotype)
train_data_precent <- 0.8
ind <- createDataPartition(genotype$Mother, p = train_data_precent, list = FALSE)
train_data <- genotype[ind,]
test_data<- genotype[-ind,]
#n <- nrow(dataset)
#train_data <- dataset[1:(n * train_data_precent),]
#test_data <- dataset[1:(n * (1 -train_data_precent)),]

#create rpart tree
library(rpart)
library(rpart.plot)
train_data_rpart <- rpart(Wt ~ ., data = train_data)
test_data_rpart <- rpart(Wt ~ ., data = test_data)
rpart.plot(train_data_rpart)

#create binary tree
#library(party)
#train_data_ctree <- ctree(Wt ~ ., data = train_data)
#test_data_ctree <- ctree(Wt ~ ., data = test_data)
#plot(train_data_ctree)


listLearners()
#create task for dataset
task <- makeClassifTask(id = 'dataset', data = genotype, target = "Mother");
lrn <- makeLearner("classif.rpart", predict.type = "prob")
rdesc = makeResampleDesc(method = "CV", stratify = TRUE)
r = resample(learner = lrn, task = task, resampling = rdesc, show.info = T)

library(C50)
library(rFerns)
library(randomForestSRC)
lrns <- makeLearners(c("lda","rpart", "C50","rFerns","randomForestSRC"),
                     type = "classif")
benchmark_result <- benchmark(learners = lrns, tasks = task, resampling = cv5)

plt = plotBMRBoxplots(benchmark_result, measure = mmce,
                      order.lrn = getBMRLearnerIds(benchmark_result))
head(plt$data[, -ncol(plt$data)])
levels(plt$data$task.id) = c("genotype")
levels(plt$data$learner.id) = c("lda","rpart", "C50","rFerns","randomForestSRC")

plt + ylab("Error rate")
```

monitors

Matryca	Rodzaj_matrycy	Rozdzielczosc_w	Rozdzielczosc_h	Jakosc_ekranu	Czas_reakcji_matrycy	Czestotliwosc_odswiezania	Kontrast_statyczny	Cena	Ocena	Liczba_ocen
18.5	TFT	1366	768	200	5	60	700	259	4	150
23.6	MVA	1920	1080	250	5	60	3000	449	4	3
24	TN	1920	1080	250	1	60	1000	499	4	114
24	VA	1920	1080	250	4	60	3000	549	4	1
23.8	IPS	1920	1080	250	1	75	1000	549	4	107
24.5	TN	1920	1080	250	1	75	1000	599	4.5	28
23.8	IPS	1920	1080	250	1	75	3000	599	4.5	45
15	IPS	2560	1080	250	5	75	1000	649	4.5	15
27	IPS	1920	1080	250	5	60	1000	699	4.5	53
27	VA	1920	1080	250	4	60	3000	749	4.5	68
27	VA	1920	1080	250	4	60	3000	749	5	77
24	TN	1920	1080	350	1	144	1000	829	5	25
27	VA	1920	1080	250	4	75	3000	1029	5	9
23.6	VA	1920	1080	300	1	144	3000	1099	5	1
24	TN	2560	1440	350	1	165	1000	1839	5	14
27	TN	2560	1440	350	1	144	1000	1999	5	13
34	IPS	3440	1440	300	5	75	1000	2999	5	14

Plik monitors.csv

3. Wynik działania:



Plik tree.png

4. Wnioski

Na podstawie otrzymanego wyniku można stwierdzić, że język R oraz paczka mlr pozwalają w prosty i wygodny sposób drzewa decyzyjne które mogą służyć do klasyfikacji. Sukces utworzenia drzewa jest uzależniony od wielkości zbioru uczącego. Paczka mlr pozwala również na uczenie sieci nauronowych. Skuteczność nauczenia sieci zależy od wprowadzonych danych.