Zajęcie 7. Modelowanie procesów uczenia maszynowego w pakecie mlr. Trenowanie, ocena i porównywanie modeli w pakecie mlr

Abstract

Celem jest uczenie maszynowe za pomocą pakitu mlr

1. Użycie drzew decyzyjnych. Metoda C5.0

Drzewa decyzyjne stanowią model decyzyjny, w którym w uporządkowany sposób przedstawia się hierarchiczne ciągi działań (w pełni zależnych od decydenta) i zdarzeń (niezależnych od decydenta, czasami o charakterze losowym). Graficzne przedstawienie w postaci drzewa decyzyjnego ułatwia analizę wszystkich elementów sytuacji istotnych przy podejmowaniu decyzji. W efekcie możliwe staje się określenie wariantów decyzyjnych i ich konsekwencji. W modelu tym nie występują tu w jawnej postaci warunki sztywne i elastyczne, są one uwzględniane w trakcie budowy drzewa. Dodatkowe podanie prawdopodobieństw i kosztów poszczególnych wariantów decyzyjnych prowadzi do zwiększenia racjonalności optymalizacyjnej poprzez maksymalizację funkcji użyteczności. Celem stosowania modelu w postaci drzewa decyzyjnego jest uproszczenie oceny sytuacji decyzyjnej, model ten pozwala na jednoczesną analizę wielu wariantów decyzyjnych i kryteriów ich oceny. Model taki jest użyteczny, o ile drzewo nie staje się zbyt obszerne (nie mieści się na kartce lub ekranie). Z wykorzystaniem drzew decyzyjnych może być prowadzona analiza wielowariantowa (what-if analysis), a poprzez implementację programową możliwe jest zastosowanie tego modelu w komputerowych systemach wspomagania decyzji.

Przykład. Przykład typowego drzewa decyzyjnego (2-poziomowego) przedstawiono na Rys. 1.

Rozważana jest tutaj sytuacja związana z ubezpieczeniem mieszkania, przy założeniu kosztów polisy w wysokości 3% oraz wkładu własnego w wysokości 2% wartości mieszkania. Możliwym zdarzeniom (brak kradzieży,

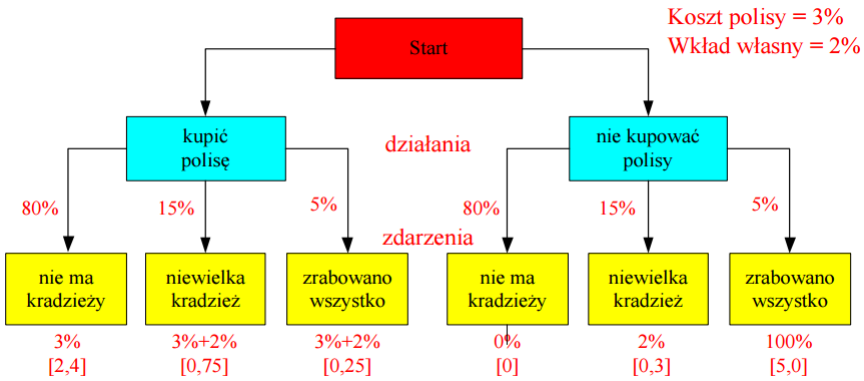


Figure 1: Drzewo decyzyjne dot. decyzji o zakupie polisy na ubezpieczenie mieszkania

kradzież niewielka – nie przekraczająca wkładu własnego oraz kradzież pełna) przypisano prawdopodobieństwa ich wystąpienia (odpowiednio 80, 15 i 5%). Zarówno wydatki związane z zakupem polisy, z ponoszeniem wkładu własnego, jak i rekompensatą za skradzione wyposażenie mieszkania (w przypadku rezygnacji z zakupu polisy) traktowane są jako strata, którą należy zminimalizować.

Wartość oczekiwaną straty związanej z daną decyzją można obliczyć z zależności (1.2) wprowadzonej przy okazji omawiania strategii scalania prawdopodobieństw i użyteczności, przy czym tutaj użytecznością (negatywną – strata) będzie koszt poniesiony przy danym wariancie decyzyjnym dla poszczególnych zdarzeń. Wartości iloczynów pi(sk) ui(sk) umieszczono w nawiasach kwadratowych pod zdarzeniami na Rys. 1. Oczekiwana strata dla poszczególnych decyzji wynosi

$$SPU(d_i) = \begin{cases} 3.4, & \text{dla decyzji kupować polisę} \\ 5.3, & \text{dla decyzji nie kupować polisę} \end{cases}$$

a zatem właściwą decyzją będzie zakup polisy ubezpieczeniowej.

2. Drzewa decyzyjne na podstawie metody CART

2.1. Paczki używane

- **readr** dla ładowania danych

- **dplyr** dla przetwarzania danych

- **party** oraz **rpart** dla algorytmów drzew klasyfikacyjnych

dplyr importuje **magrittr** który używa składnię "%>%" poniżej. Więcej informacji jest w Google.

2.2. Wybieramy atrybuty którzy mogą spowodować rozbicie danych

Dla przykładu:
każdy pasażer jest reprezentowany przez wiersz danych, kolumny są atrybuty

- **survived**: 0 if died, 1 if survived

- **embarked:** Port of Embarkation (Cherbourg, Queenstown,Southampton)

- **sex:** Gender

- **sibsp:** Number of Siblings/Spouses Aboard

- **parch:** Number of Parents/Children Aboard

- **fare:** Fare Payed

Pozostałe kolumny mogą być używane w innych zagadnieniach

2.3. Przed ustaleniem modelu atrybuty kategoriowane mają być przekształcani w faktorzy

Dla przykładu:

```
titanic3 <- "https://goo.gl/At238b" %>%
 read_csv %>% # read in the data
 select(survived, embarked, sex,
        sibsp, parch, fare) %>%
 mutate(embarked = factor(embarked),
        sex = factor(sex))
```

2.4. Rozbijanie danych dla nauczania i testowania

Dla przykładu:

```
.data <- c("training", "test") %>%
 sample(nrow(titanic3), replace = T) %>%
 split(titanic3, .)
```

2.5. Rozbicie rekursyjne jest zrealizowane w paczce "rpart"

Dla przykładu:

```
rtree_fit <- rpart(survived ~ .,
                   .data$training)
rpart.plot(rtree_fit)
```

2.6. Warunkowe rozbicie jest zrealizowane przez metodę "ctree"

Dla przykładu:

```
tree_fit <- ctree(survived ~ .,
                  data = .data$training)
```

3. Reguły klasyfikacyjne

```
ruleModel <- C5.0(churn ~ ., data = churnTrain, rules = TRUE)
ruleModel
summary(ruleModel)
```

4. Pakiet mlr

- dokumentacja:

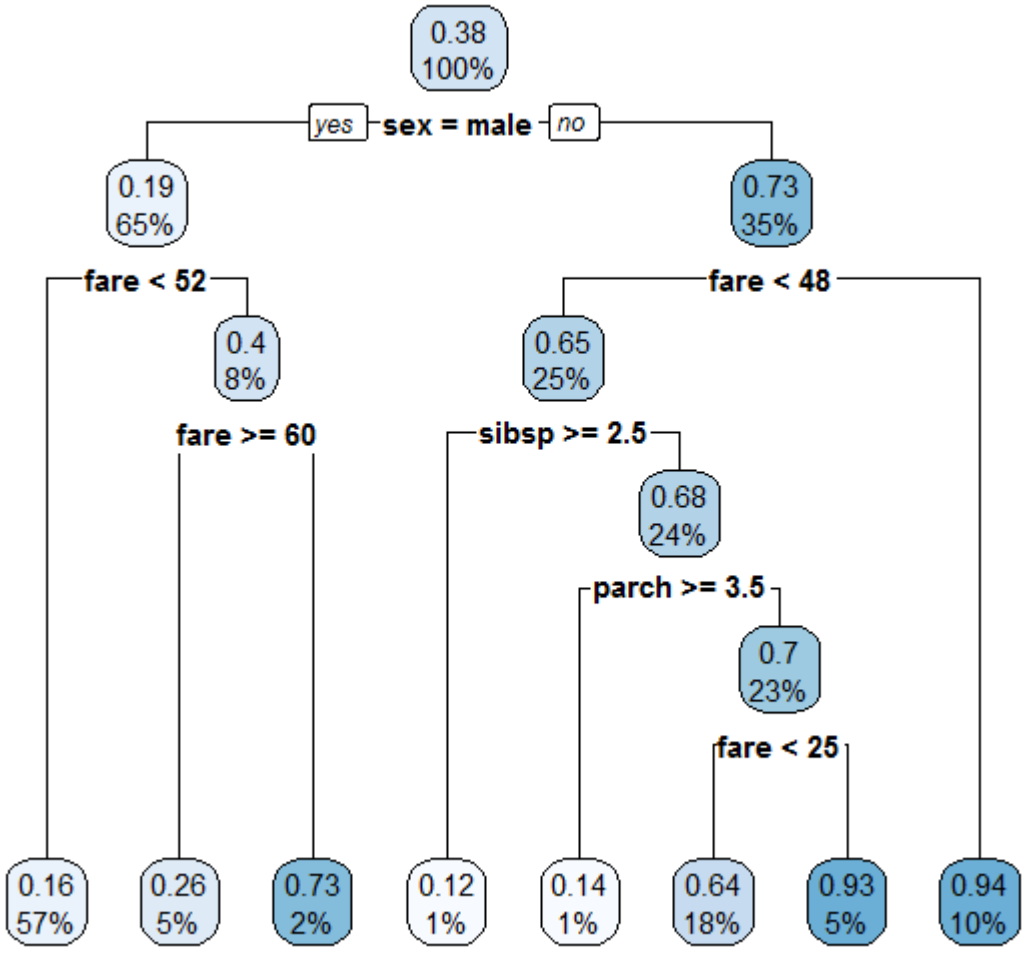
  `https://cran.r-project.org/web/packages/mlr/vignettes/mlr.html`

- dokumentacja na CRAN:

  `https://cran.r-project.org/web/packages/mlr/mlr.pdf`

- "tutorial":

  https://www.analyticsvidhya.com/blog/ 2016/08/practicing-machine-learning-techniques-in-r-with-mlr-package/


0.38
100%
yes
sex = male
no
0.19
65%
0.73
35%
fare < 52
fare < 48
0.4
8%
0.65
25%
fare >= 60
sibsp >= 2.5
0.68
24%
parch >= 3.5
0.7
23%
fare < 25
0.16
57%
0.26
5%
0.73
2%
0.12
1%
0.14
1%
0.64
18%
0.93
5%
0.94
10%

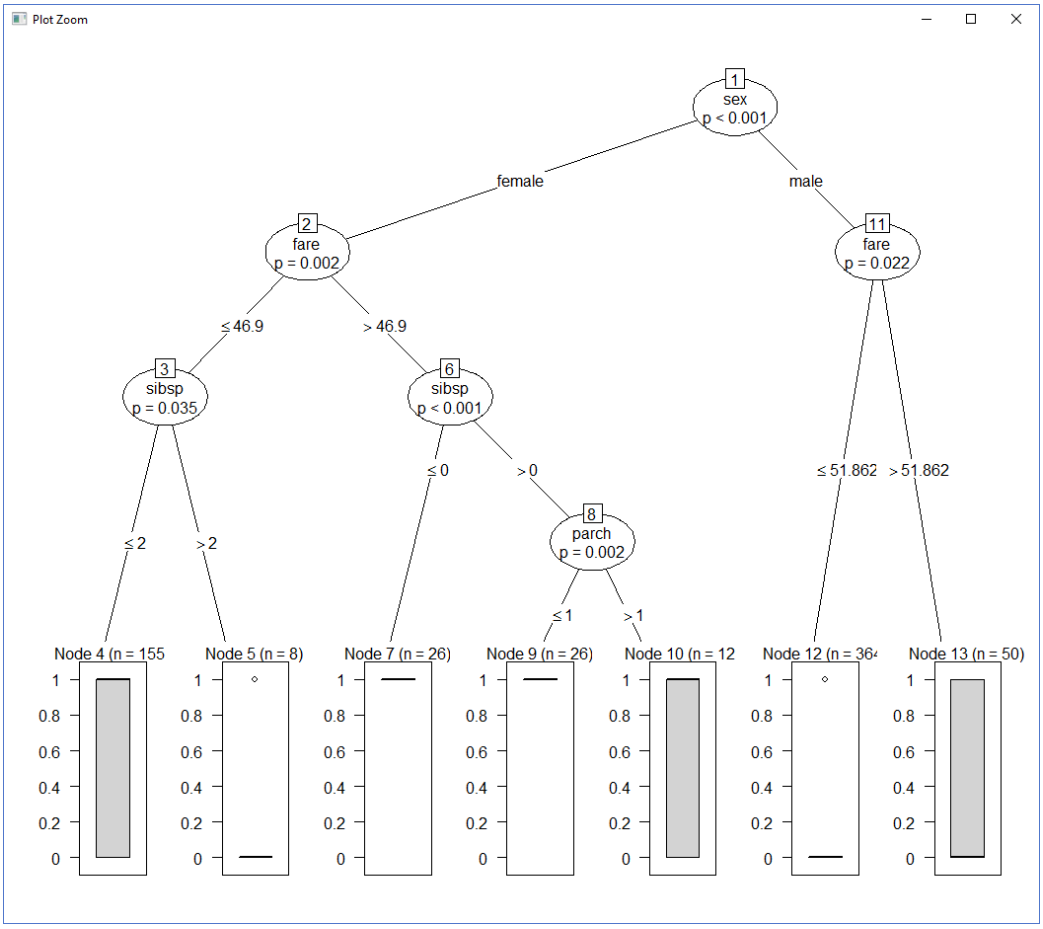


Figure 2: Użycie plot

- wyświetlenie wszystkich możliwych learnerów

```
> listLearners()
```

- wyświetlenie typów możliwych learnerów

```
> levels(factor(listLearners()$type))
[1] "classif"    "cluster"    "multilabel" "regr"   "surv"
```

- definiowanie zadania

```
## Define the task
task =
makeClassifTask(id = deparse(substitute(data)), data, target,
  weights = NULL, blocking = NULL, coordinates = NULL,
  positive = NA_character_, fixup.data = "warn", check.data = TRUE)
makeClusterTask(id = deparse(substitute(data)), data, weights = NULL,
  blocking = NULL, coordinates = NULL, fixup.data = "warn",
  check.data = TRUE)

makeCostSensTask(id = deparse(substitute(data)), data, costs,
  blocking = NULL, coordinates = NULL, fixup.data = "warn",
  check.data = TRUE)

makeMultilabelTask(id = deparse(substitute(data)), data, target,
  weights = NULL, blocking = NULL, coordinates = NULL,
  fixup.data = "warn", check.data = TRUE)

makeRegrTask(id = deparse(substitute(data)), data, target,
  weights = NULL, blocking = NULL, coordinates = NULL,
  fixup.data = "warn", check.data = TRUE)

makeSurvTask(id = deparse(substitute(data)), data, target,
  weights = NULL, blocking = NULL, coordinates = NULL,
  fixup.data = "warn", check.data = TRUE)
```

- metoda próbkowania

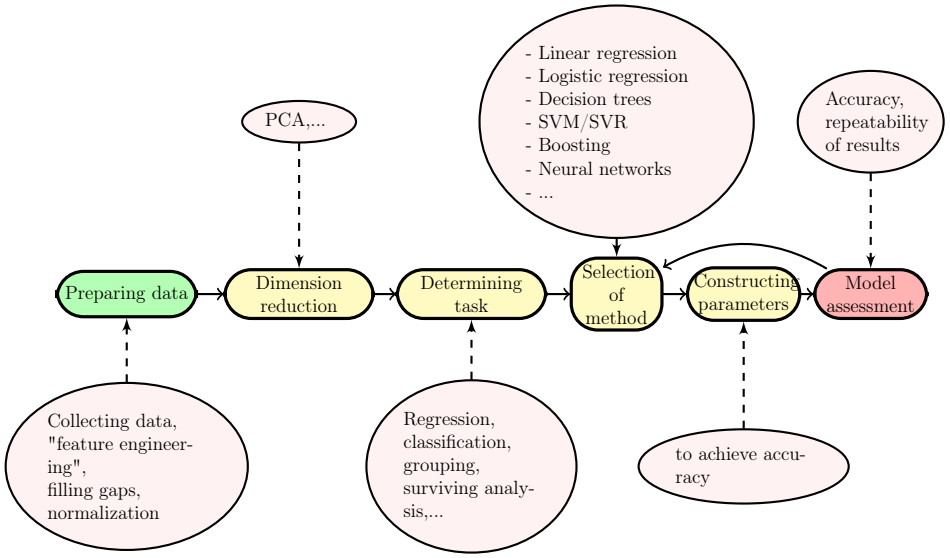


Figure 3: Development of machine learning model

```
## Define the resampling strategy
rdesc = makeResampleDesc(method = "CV", stratify = TRUE)
## CV - cross validation
```

- "uczenie"

```
## Do the resampling
r = resample(learner = lrn, task = task, resampling = rdesc,
             show.info = FALSE)
```

- "uczenie"-porównanie kilku learnerów

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

5. Warianty Zadania

Uwaga! Sprawozdania muszą być sporządzane zgodnie ze wzorem. Oprócz tego pliki źródłowe oraz obrazy muszą być zachowane w zdalnym repozytorium.

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

1. iris
2. infert
3. mtcars
4. Aids2
5. bacteria
6. biopsy
7. cats
8. genotype
9. shuttle
10. Pima.tr2
11. OME
12. Melanoma

Zadanie 2. Zadanie 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.

References