A Gentle Tutorial on Decision Trees
Using `rpart()` implementation in R

NPFL 054 Lab session (Hladká and Holub, 2017)

http://ufal.mff.cuni.cz/course/npfl054

In this tutorial we will work with 'Forbes2000' dataset, which is a part of HSAUR library in R. Forbes2000 data set lists a ranking of the world's biggest companies, measured by sales, profits, assets and market value. To get more info about Forbes2000 data set, use `help(Forbes2000)`.

We will try to predict profit of the companies using Decision Trees model.

Part I The data set – elementary exploration

```
> library(HSAUR)                        # loading the library with Forbes data set

> str(Forbes2000)                       # structure of the data set

'data.frame':2000 obs. of  8 variables:
 $ rank       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ name       : chr  "Citigroup" "General Electric" "American Intl Group" "ExxonMobil" ...
 $ country    : Factor w/ 61 levels "Africa","Australia",..: 60 60 60 60 56 60 56 28 60 ...
 $ category   : Factor w/ 27 levels "Aerospace & defense",..: 2 6 16 19 19 2 2 8 9 20 ...
 $ sales      : num  94.7 134.2 76.7 222.9 232.6 ...
 $ profits    : num  17.85 15.59 6.46 20.96 10.27 ...
 $ assets     : num  1264 627 648 167 178 ...
 $ marketvalue: num  255 329 195 277 174 ...


> head(Forbes2000)                            # displays the beginning of the data
> levels(Forbes2000$category)                 # look at the categories

# ***************************************************
# How many companies are reporting positive profit?
# ***************************************************

> table(Forbes2000$profits > 0)
FALSE  TRUE
  290  1705

> table(Forbes2000$profits < 0)
FALSE  TRUE
 1715   280

> table(Forbes2000$profits == 0)
FALSE  TRUE
 1985    10

# Observation: 1705 + 280 + 10 does not sum up to the total data set size!
> nrow(Forbes2000)
[1] 2000
```

```
# ... because there are some missing values
> table(is.na(Forbes2000$profits))
FALSE  TRUE
 1995     5

# *******************************************
# Data transformation
# *******************************************

> F = Forbes2000                              # just to make a copy

# for simplicity, NA values will be replaced by zeros, and then
# the numerical values 'profits' will be reduced to a binary attribute

> F$profits[is.na(F$profits)] = 0             # NAs are replaced by 0
> F$profits = factor(F$profits > 0.2)         # transformation to a binary variable
> table(F$profits)

FALSE  TRUE
 1044   956
```

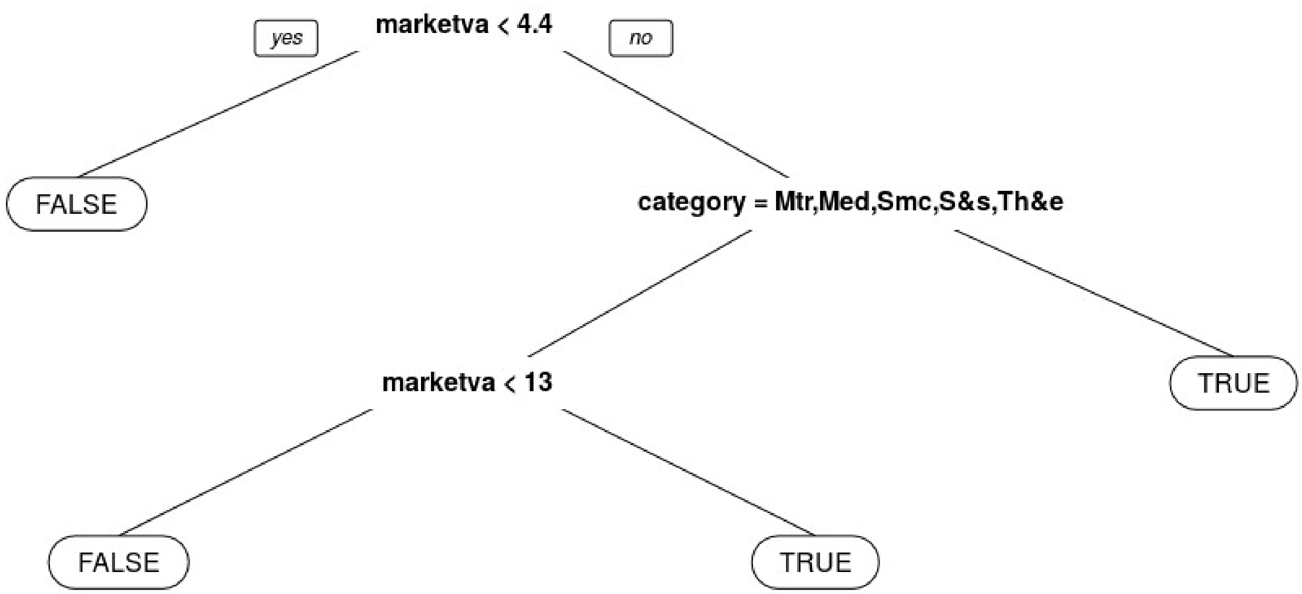
Part II Building a simple Decison Tree

Now we will do a binary classification task. The binary value '`profit`' will be predicted using one categorical feature (`category`) and three numerical features `(sales, assets, marketvalue).`

```
# *******************************************
# TRAINING AND BASIC EVALUATION
# *******************************************

# to randomly split the data into two disjoint subsets
> set.seed(123); s = sample(2000)
> data.train = s[1:1000]                        # indices of training examples
> data.test  = s[1001:2000]                     # indices of test examples
> length(unique(c(data.train,data.test)))       # just to check

# to train the model/hypothesis using the train data set
> library(rpart)
> forbes.train = F[data.train, 1:8]
> model = rpart(profits ~ category + sales + assets + marketvalue, forbes.train)

# to display the resulting DT
> plot(model, branch=1, uniform=T); text(model, use.n=T,  digits=3, all=T)

# a nicer visualisation
> library(rpart.plot)
> rpart.plot(model)

# to evaluate the prediction using the test set
> forbes.test = F[data.test, 1:8]
> prediction <- predict(model, forbes.test, type="class")
> table(prediction)

# simple evaluation
> model.cm = table(prediction, forbes.test$profits)              # confusion matrix
> message("Accuracy = ", sum(diag(model.cm))/1000)
```

Details of the decision tree structure are described in the model

```
> print(model)
n= 1000

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 1000 497 FALSE (0.5030000 0.4970000)
   2) marketvalue< 4.44 450  87 FALSE (0.8066667 0.1933333) *
   3) marketvalue>=4.44 550 140 TRUE (0.2545455 0.7454545)
     6) category=Materials,Media,Semiconductors,
        Software & services,Technology hardware & equipment
        85  37 FALSE (0.5647059 0.4352941)
      12) marketvalue< 13.2 53  13 FALSE (0.7547170 0.2452830) *
      13) marketvalue>=13.2 32   8 TRUE (0.2500000 0.7500000) *
     7) category=Aerospace & defense,Banking,
        Business services & supplies,
        Capital goods,Chemicals,Conglomerates,Construction,
        Consumer durables,Diversified financials,
        Drugs & biotechnology,Food drink & tobacco,
        Food markets,Health care equipment & services,
        Hotels restaurants & leisure,
        Household & personal products,Insurance,
        Oil & gas operations,Retailing,
        Telecommunications services,
        Trading companies,Transportation,Utilities
        465  92 TRUE (0.1978495 0.8021505) *
```



Part III Learning parameters

Exercises

1. Try and play with parameters of `rpart()` and observe how the prediction accuracy changes.

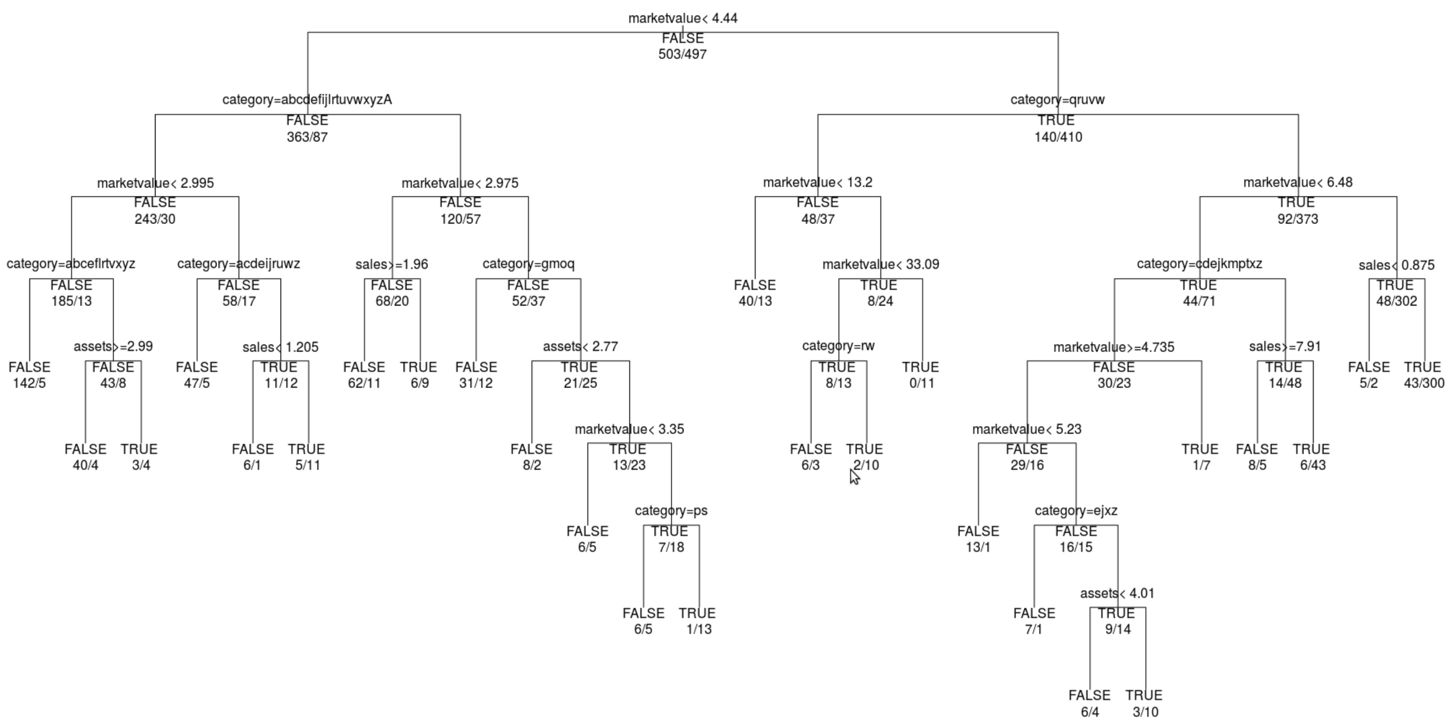
2. Use additional feature '`country`' (categorical variable). Does it help to predict profits?

3. Build a deeper tree by setting cp

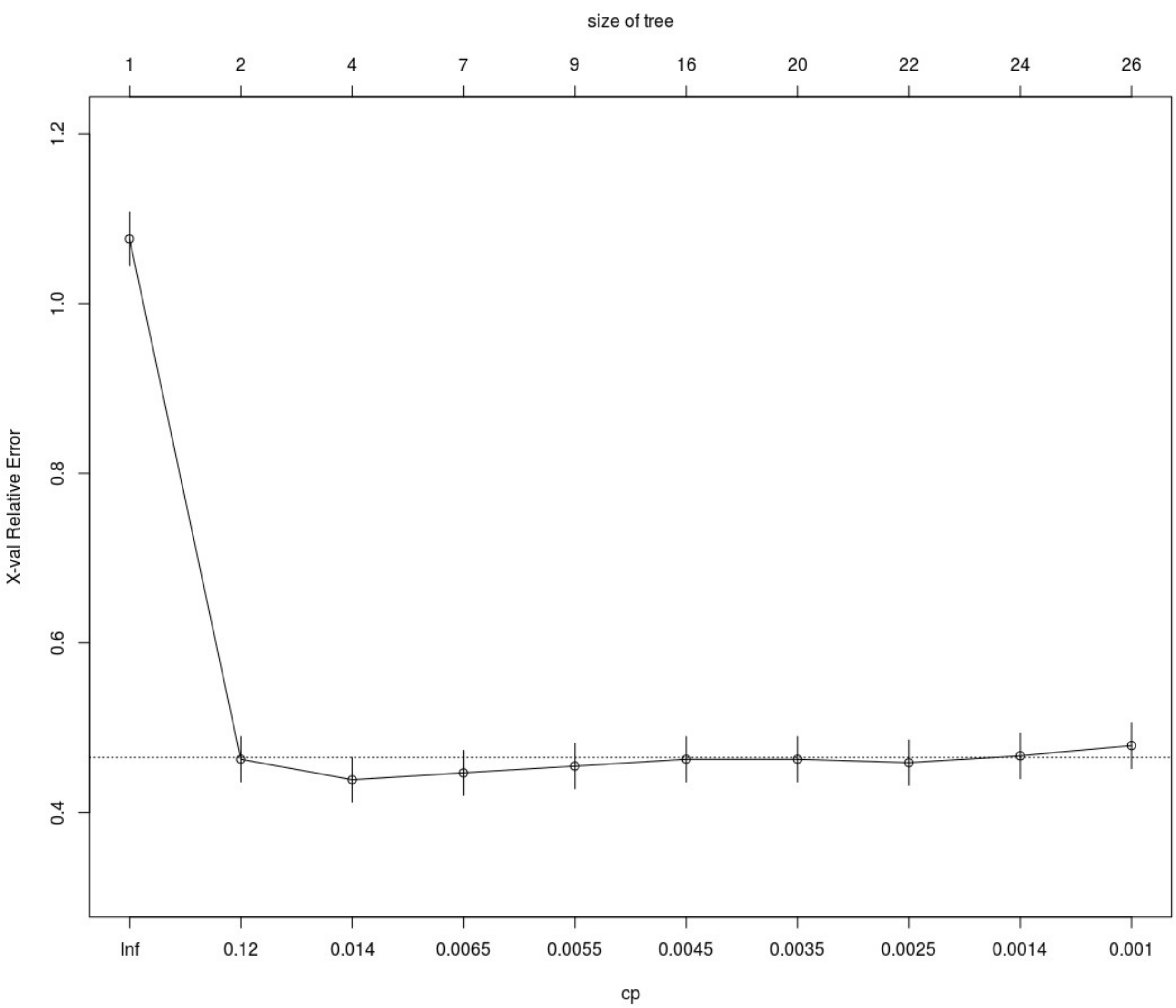
```
M.deep = rpart(profits ~ category + sales + assets + marketvalue, forbes.train,
               cp = 0.001)
plot(M.deep, branch=1, uniform=T); text(m, use.n=T, digits=3, all=T)
```

– Is this deeper model better?

```
plot(M.deep, branch=1, uniform=T); text(m, use.n=T,  digits=3, all=T)
```



```
# see the cp values
printcp(M.deep)

Root node error: 497/1000 = 0.497

          CP nsplit rel error  xerror     xstd
1  0.5432596      0   1.00000 1.07646 0.031736
2  0.0271630      1   0.45674 0.46278 0.026776
3  0.0070423      3   0.40241 0.43863 0.026271
4  0.0060362      6   0.37626 0.44668 0.026443
5  0.0050302      8   0.36419 0.45473 0.026611
6  0.0040241     15   0.32797 0.46278 0.026776
7  0.0030181     19   0.31187 0.46278 0.026776
8  0.0020121     21   0.30584 0.45875 0.026694
9  0.0010060     23   0.30181 0.46680 0.026858
10 0.0010000     25   0.29980 0.47887 0.027096

plotcp(M.deep)
```



Compare the test results

```
# prediction using the original model
> prediction.model = predict(model, forbes.test, type="class")
> table(prediction.model)

prediction
FALSE  TRUE
  505   495

> model.cm = table(prediction.model, forbes.test$profits)
> model.cm

prediction FALSE TRUE
     FALSE   405  100
     TRUE    136  359

> message("Accuracy = ", sum(diag(model.cm))/1000)
Accuracy = 0.764
```

```
# prediction using M.deep
> prediction.M.deep = predict(M.deep, forbes.test, type="class")
> table(prediction.M.deep)

prediction.M.deep
FALSE  TRUE
  511   489

# confusion matrix
> M.deep.cm = table(prediction.M.deep, forbes.test$profits)
> M.deep.cm

prediction.M.deep FALSE TRUE
            FALSE   406  105
            TRUE    135  354

> message("Accuracy = ", sum(diag(M.deep.cm))/1000)
Accuracy = 0.76
```