R course - Exercice 3

First Name - Last Name

Master 2 Statistics and Econometrics

Problem

We consider the following simulated data set of size $n = 1000$ observations and $p = 10001$ variables.

```
n_simu <- 1000
set.seed(1)
df_simulate <- data.frame(x_1 = rnorm(n_simu))
for (k in 2:10000) {
set.seed(k)
df_simulate[, paste0("x_", k)] <- rnorm(n_simu)
}
df_simulate[, "y"] <- runif(n_simu, 0, 0.5)
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 > 0.8, "y"] <-
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 > 0.8, "y"] + 5.75
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 <= 0.8 & df_simulate$x_30 > 0.5, "y"] <-
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 <= 0.8 & df_simulate$x_30 > 0.5, "y"] + 18.95
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 <= 0.8 & df_simulate$x_30 <= 0.5, "y"] <-
df_simulate[df_simulate$x_40 > 0 & df_simulate$x_99 <= 0.8 & df_simulate$x_30 <= 0.5, "y"] + 20.55
df_simulate[df_simulate$x_40 <= 0 & df_simulate$x_150 < 0.5, "y"] <-
df_simulate[df_simulate$x_40 <= 0 & df_simulate$x_150 < 0.5, "y"] - 5
df_simulate[df_simulate$x_40 <= 0 & df_simulate$x_150 >= 0.5, "y"] <-
df_simulate[df_simulate$x_40 <= 0 & df_simulate$x_150 >= 0.5, "y"] - 10
```

Let y be the dependent variables and $x_1, ..., x_{10000}$ the explanatory variables. When using function *rpart()* in package **rpart**, we obtain the following regression tree:

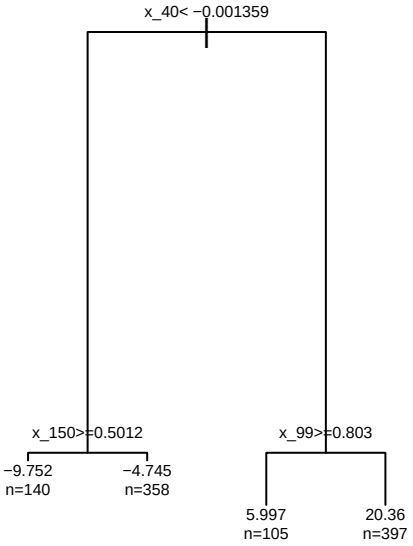
```
require(rpart)
```

```
## Loading required package: rpart
```

```
par(mfrow = c(1, 2), xpd = NA)
system.time(
fit <- rpart(y ~ ., data = df_simulate,
control = rpart.control(maxdepth = 2))
)
```

```
##    user  system elapsed
##  15.084   2.036  15.657
```

```
plot(fit)
text(fit, use.n = TRUE, cex = 0.5)
```



The objectives of this exercice are:

- program your own functions to reproduce more or less similar results to those obtained with the package **rpart**.
- try to improve computational times by using parallel computing.

Bibliography on the regression trees:

- in French: https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-cart.pdf
- In English: https://web.stanford.edu/~hastie/Papers/ESLII.pdf (p. 307)

Hypotheses

- we suppose that all variables are **numeric**
- the deep of the trees is equal to 2

Part 1

Program the function *split_noeud()* that takes as input arguments:

- a vector **y** of size **n**,
- a vector **x** of the same size as **y**,
- a scalar **n_min** equal to 5 by default.

The purpose of the *split_noeud()* function is to find the best possible value, noted **x_split**, among the values of **x**, which separates the starting sample into 2 subsamples (each sample must have a size greater than or equal to **n_min**): a sample of size n_1 such as observations $x_1^1, \dots x_{n_1}^1 \leq$ **x_split** and a sample of size n_2 such as **x_split** $< x_1^2, \dots x_{n_2}^2$. **x_split** is chosen so that the sum of squares error (SSE) (calculated on the variable to explain **y** and equal to $(\sum_{i=1}^{n_1}(y_i^1 - \bar{y}_1)^2 + \sum_{i=1}^{n_2}(y_i^2 - \bar{y}_2)^2)$) is minimum.

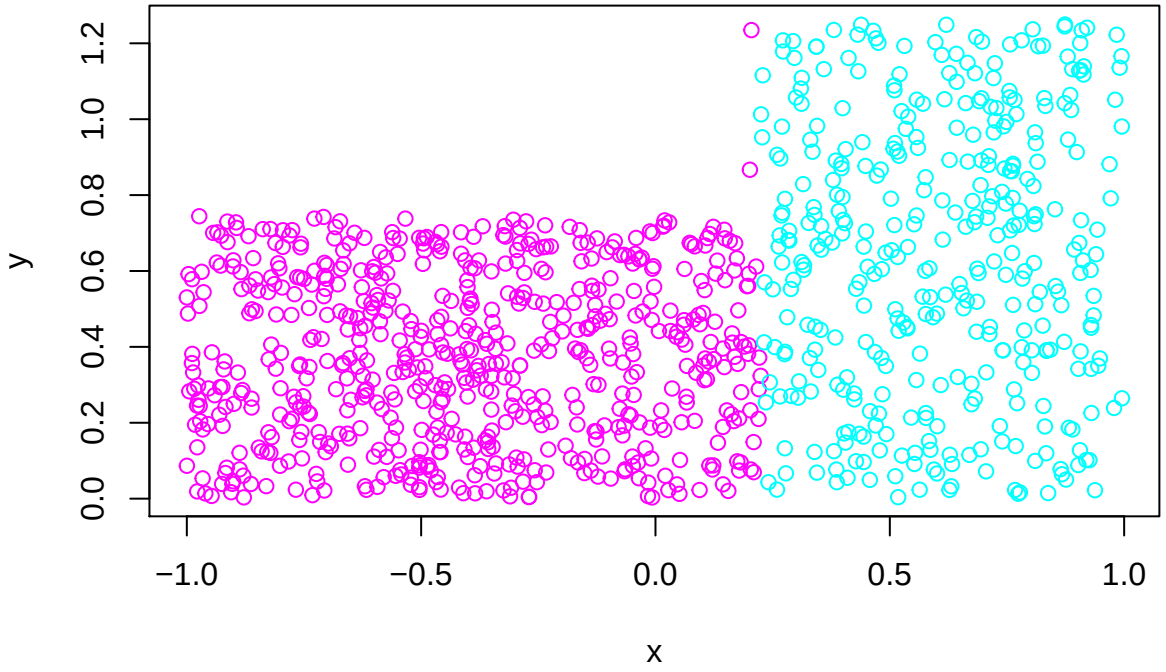
split_noeud() returns a **list** with 2 elements:

- **x_split**, which corresponds to the value of **x** which permits to cut the sample into two sub-samples with respect to the rule given above.
- **var_intra**, the value of the intra-variance of y when spliting the sample into 2 subsamples with respect to **x_split**.

Exemple of application :

```r
x <- runif(1000, -1, 1)
y <- runif(1000) * ifelse(x > 0.2, 1.25, 0.75)
(res_example <- split_noeud(y, x))
```

```
## $x_split
## [1] 0.2245128
##
## $var_intra
## [1] 78.2122
```



Part 2

- By using *lapply()* (or *sapply()*), find the variable among all the explanatory variables included in **df_simulate** which permits to get the intra-variance the smallest. Which is the corresponding value of **x_split** ?
- do the same thing by using paralle computing and compare the computation time between non parallel and parallel version.

Nodes 2 and 3

Let **ech_2** and **ech_3** be the two sub-samples (or nodes) obtained by splitting the initial sample with respect to what has been done in the previous step.

- By doing parallel computing, find in the node 2 (respect. node 3) which is the variable and corresponding value **x_split** which permit to separate the node 2 in an optimal way.

Comparison

Compare the computational time obtained between your codes and the function *rpart()*.

Notation

You will have to return the exercice in *.pdf* or *.html* format, which would have been done with **R** Markdown if possible. It should contain the lines of code used to answer the questions, but you should also explain what you are doing.