Instituto Superior de Engenharia de Lisboa
Mestrado em Engenharia Informática e de Computadores
Mestrado em Engenharia Informática e Multimédia
Mestrado em Matemática Aplicada para a Indústria

Big data mining (MDLE)

Laboratory Class #1 — R language and environment setup
2^{nd} semester, 2022/2023 (March, 15)

Code and Report are due by March, 22

1. **Data Resources and Software Tools.**
   For this laboratory class, you will need the following software:

   - Access to RServer (`http://datalys.dyn.fil.e.ipl.pt:8787`) or,
   - R, https://cran.r-project.org/, and
   - R Studio, https://posit.co/downloads/.

   As a preparation to the lab, you should:

   - download the Sparklyr and RStudio IDE cheat sheets from https://posit.co/resources/cheatsheets/.

2. **R Primer**.

   (a) Go to http://www.r-tutor.com/r-introduction and read the basic data types section.

   (b) Open R Studio, execute and explain each of the following statements, one by one (and check the values on the 'Global Environment' tab). Beware with the cut+paste:

   i. Scalars and Operators.

```
var1 <- 3
show(var1)
var2 <- var1 * var1
var3 <- var1 ** 2
var4 <- var1 ^ 2
var1 < var1
var3 != var4
var2 == var4
var2 <- var2 - var2
var5 <- var3 / var2
var5 + 1
```

   ii. Manipulating vectors.

```
a <- c(1,2,5.3,6,-2,4,3.14159265359)
a
b <- c("1","2","3")
"2" %in% b
"5" %in% b
```

```
c <- c(TRUE,TRUE,FALSE,TRUE)
a <- c
a[0]
a[1]
a[-1]
a[8]
a[c(1,3)]
a[c(3,1)]
a[a > 2]
```

iii. Manipulating matrices.

```
m <- matrix(1:6, nrow=3,ncol=2)
show(m)
n <- matrix(2:7, nrow=2,ncol=3)
n
m[,2]
n[1,]
m[2:3,1:2]
n %*% m
m %*% n
n %*% n
n^2
sqrt(n)
```

iv. Manipulating data frames.

```
d <- c(1,2,3,4)
e <- c("Bob", "Alice", NA,"Joe")
f <- c(TRUE,TRUE,FALSE,TRUE)
my.data <- data.frame(d,e,f,stringsAsFactors=FALSE)
show(my.data)
names(my.data) <- c("ID","Name","Passed")
View(my.data)
my.data$Name
my.data <- cbind(my.data, Failed=!f)
View(my.data)
my.data <- rbind(my.data, c(5,"Carol",FALSE,TRUE))
View(my.data)
?data.frame
```

v. Manipulating factors.

```
colour <- c(rep("red",20), rep("blue", 30))
colour <- factor(colour)
summary(colour)
dimensions <- c("large", "medium", "small")
show(dimensions)
dimensions <- ordered(dimensions)
show(dimensions)
```

vi. Some useful functions.

```
length(colour)
length(a)
class(colour)
class(dimensions)
class(a)
class(my.data)
nrow(my.data)
ncol(my.data)
str(my.data)
sessionInfo()
ls()
rm(a)
glimpse(my.data)
```

vii. Packages.

```
library()
search()
library(”MASS”)
search()
.libPaths()
install.packages(”e1071”)
install.packages(”funModeling”)

x<-c(”MASS”,”dplyr”, ”e1071”)
lapply(x, require, character.only = TRUE)

require(funModeling)
```

viii. Basic descriptive statistics and more.

```
iris
summary(iris)
fivenum(iris$Sepal.Length)

status(iris)
```

(c) Open the file “**ControlFlow.R**”, handed with this guide, and:

i. Set a breakpoint at line 6 and run the code delimited by region *EX.1*, step by step.

ii. Explain the purpose of the three examples, named *EX.1*, *EX.2*, and *EX.3*.

(d) Implement and show a code that gets the first two columns of *my.data*, using the range operator indexation.

(e) Implement and show a code that gets the first two columns of *my.data*, using vector indexation.

(f) Install the package `Hmisc`, and use the function `describe` on *my.data* variable.

3. **Setup Spark**.

(a) Install the package `sparklyr` using `install.packages("sparklyr")`. It may take some minutes, so be patient.

(b) Install *Spark* using `library(sparklyr) spark_install(version = ’3.3.2’, hadoop_version = ’3’)`

4. **Spark primer**.
Describe the commands used (if not listed), the problems (if any), and the results of the following points, line by line:

(a) Check, programmatically, if `sparklyr` package is loaded. Next, connect to *Spark* using `ss <- spark_connect(’local’, version = ’3.3.2’, hadoop_version = ’3’, config = list())`.

(b) View the content of variable `ss`.

(c) Use function `copy_to` from package `dplyr` to copy the *iris* data set to *Spark*.

```
library(dplyr)
df <- copy_to(ss, iris)
show(df)
```

(d) Show some sample data from the loaded data set.

```
head( select( df, Petal_Width, Species))
head( filter( df, Petal_Width > 0.3))
df %>% head
```

(e) Using SQL

```
library(DBI)
df_sql <- dbGetQuery(ss, ”SELECT * FROM iris WHERE Petal_Width > 0.3 LIMIT 5”)
show(df_sql)
```

(f) Get data from *Spark* nodes.

```
local_df <- collect(df)
show(local_df)
show(df)
```

(g) Disconnect spark using `spark_disconnect(ss)`. Confirm, programmatically, that *Spark* is closed.

(h) Indicate, as an example, a supervised learning algorithm implemented on *Spark*, and available through `sparklyr`.

(i) Indicate, as an example, a unsupervised learning algorithm implemented on *Spark*, and available through `sparklyr`.

(j) Tell how *Spark* can support the *Big data process pipeline*. Try to start at https://spark.rstudio.com