Practical Exam - Exercise #1

Marivic Amuan

2024-03-07

A.LOAD THE BUILT-IN WARPBREAKS DATASET

1. Find out, in a single command, which columns of warpbreaks are either numeric or integer. What are the data types of each column?

Answer: The data types of each column are integer and factor.

```
warpbreaks
```

```
##    breaks wool tension
## 1      26    A       L
## 2      30    A       L
## 3      54    A       L
## 4      25    A       L
## 5      70    A       L
## 6      52    A       L
## 7      51    A       L
## 8      26    A       L
## 9      67    A       L
## 10     18    A       M
## 11     21    A       M
## 12     29    A       M
## 13     17    A       M
## 14     12    A       M
## 15     18    A       M
## 16     35    A       M
## 17     30    A       M
## 18     36    A       M
## 19     36    A       H
## 20     21    A       H
## 21     24    A       H
## 22     18    A       H
## 23     10    A       H
## 24     43    A       H
## 25     28    A       H
## 26     15    A       H
## 27     26    A       H
## 28     27    B       L
## 29     14    B       L
## 30     29    B       L
## 31     19    B       L
```

```
## 32     29    B       L
## 33     31    B       L
## 34     41    B       L
## 35     20    B       L
## 36     44    B       L
## 37     42    B       M
## 38     26    B       M
## 39     19    B       M
## 40     16    B       M
## 41     39    B       M
## 42     28    B       M
## 43     21    B       M
## 44     39    B       M
## 45     29    B       M
## 46     20    B       H
## 47     21    B       H
## 48     24    B       H
## 49     17    B       H
## 50     13    B       H
## 51     15    B       H
## 52     15    B       H
## 53     16    B       H
## 54     28    B       H
```

```
wae <- sapply(warpbreaks,function (x) paste(class (x), collapse = "/"))
wae
```

```
##    breaks      wool   tension
## "numeric"  "factor"  "factor"
```

2. How many observations does it have?

ANSWER: It has 54 observations.

```
wae2 <- nrow(warpbreaks)
wae2
```

```
## [1] 54
```

3. Is numeric a natural data type for the columns which are stored as such? Convert to integer when necessary.

ANSWER: Yes, based on the output - the numeric shown are natural data type.

```
warpbreaks$breaks <- (warpbreaks$breaks)
warpbreaks$breaks
```

```
##  [1] 26 30 54 25 70 52 51 26 67 18 21 29 17 12 18 35 30 36 36 21 24 18 10 43 28
## [26] 15 26 27 14 29 19 29 31 41 20 44 42 26 19 16 39 28 21 39 29 20 21 24 17 13
## [51] 15 15 16 28
```

4. Error messages in R sometimes report the underlying type of an object rather than the user-level class. Derive from the following code and error message what the underlying type. Explain what is the error all about. Do not just copy the error message that was displayed.

ANSWER:

B. LOAD THE exampleFile.txt 1. Read the complete file using readLines.

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
path <- ("exampleFile.txt")
```

```
read <- readLines(path)
```

```
## Warning in readLines(path): incomplete final line found on 'exampleFile.txt'
```

```
read
```

```
## [1] "// Survey data. Created : 21 May 2013"
## [2] "// Field 1: Gender"
## [3] "// Field 2: Age (in years)"
## [4] "// Field 3: Weight (in kg)"
## [5] "M;28;81.3"
## [6] "male;45;"
## [7] "Female;17;57,2"
## [8] "fem.;64;62.8"
```

2. Separate the vector of lines into a vector containing comments and a vector containing the data. Hint: use grepl.

```
comments <- read[grepl("^//", read)]
comments
```

```
## [1] "// Survey data. Created : 21 May 2013"
## [2] "// Field 1: Gender"
## [3] "// Field 2: Age (in years)"
## [4] "// Field 3: Weight (in kg)"
```

```
dateLine <- read[!grepl("^//", read)]
dateLine
```

```
## [1] "M;28;81.3"      "male;45;"       "Female;17;57,2" "fem.;64;62.8"
```

3. Extract the date from the first comment line and display on the screen "It was created data."

```
date <- "21 May 2013"
cat("It was created data: ", date)
```

```
## It was created data:  21 May 2013
```

4. Read the data into a matrix as follows. A. Split the character vectors in the vector containing data lines by semicolon (;) using strsplit.

```
splitData <- strsplit(dateLine, ";")
splitData
```

```
## [[1]]
## [1] "M"    "28"   "81.3"
##
## [[2]]
## [1] "male" "45"
##
## [[3]]
## [1] "Female" "17"     "57,2"
##
## [[4]]
## [1] "fem." "64"   "62.8"
```

B. Find the maximum number of fields retrieved by split. Append rows that are shorter with NA's.

```
maxFields <- max(sapply(splitData, length))
maxFields
```

```
## [1] 3
```

```
row <- lapply(splitData, function(x) c(x, rep(NA, maxFields - length(x))))
row
```

```
## [[1]]
## [1] "M"    "28"   "81.3"
##
## [[2]]
## [1] "male" "45"   NA
##
## [[3]]
## [1] "Female" "17"     "57,2"
##
## [[4]]
## [1] "fem." "64"   "62.8"
```

C. Use unlist and matrix to transform the data to row-column format.

```
Data <- unlist(row)

dataMatrix <- matrix(Data, ncol = 4, nrow= 3)
dataMatrix
```

```
##      [,1]   [,2]   [,3]     [,4]
## [1,] "M"    "male" "Female" "fem."
## [2,] "28"   "45"   "17"     "64"
## [3,] "81.3" NA     "57,2"   "62.8"
```

d. From comment lines 2-4, extract the names of the fields. Set these as colnames for the matrix you just created.

```
fields <- comments[2:4]
fieldNames <- gsub("//", "", fields)

fieldNames
```

```
## [1] " Field 1: Gender"          " Field 2: Age (in years)"
## [3] " Field 3: Weight (in kg)"
```

```
rownames(dataMatrix) <- fieldNames
print(dataMatrix)
```

```
##                           [,1]   [,2]   [,3]     [,4]
##  Field 1: Gender          "M"    "male" "Female" "fem."
##  Field 2: Age (in years) "28"   "45"   "17"     "64"
##  Field 3: Weight (in kg) "81.3" NA     "57,2"   "62.8"
```