Homework 2 - Burkan Bereketoglu

2355170 - Abdullah Burkan Bereketoglu

12/23/2021

Contents

Introduction

Welcome to my homework!!!

In this homework I, Burkan, I prepared a nice pdf to show all the parts of the questions in different parts and so.

Question 1

In this question we are going to read through some files then analyze their rows, do five-number summary statistics and so.

Question 1 - part A

Here from us it is asked that to read some data files that are added to odtuclass with the homework. Data1 has variable names AGE, FAT, GENDER. Data2 has ID's MT1 scores up to 2 decimanls and also MT2 and Final scores of some identities. Lastly Data3 has bunch of values for now with some of them are starred.

```
data_1 <- read.delim("E:/2021-2022 Fall - Spring Books/STAT291/data1.txt",
                     header = FALSE,
                     stringsAsFactors = FALSE)

data_2 <- read.csv("E:/2021-2022 Fall - Spring Books/STAT291/data2.csv",
                   header = TRUE,
                   dec = ".")

data_3 <- read.delim("E:/2021-2022 Fall - Spring Books/STAT291/data3.txt",
                     header= FALSE,
                     stringsAsFactors = FALSE)


#First 5 rows of datas
head(data_1, n = 5)
```

```
##                                                    V1
## 1 Variable names are AGE, FAT, GENDER respectively,
## 2                     GENDER is a Factor variable.
## 3                                          23  19.2   m
## 4                                          28  16.6   f
## 5                                          38  32.5   f
```

```
head(data_2, n = 5)
```

```
##   ID   MT1   MT2 Final
## 1  1 71,27 79,64 95,49
## 2  2 58,86 46,61 40,24
## 3  3 74,02 39,86 66,73
## 4  4 93,38 49,25 77,47
## 5  5 68,43  51,3 72,23
```

```
head(data_3, n = 5)
```

```
##                 V1
## 1 5.2 2.7 3.9 1.4
## 2 7.3 2.9 6.3 1.8
## 3   5.7 2.6 3.5 1
## 4 4.4 3.2 1.3 0.2
## 5   5.7 3 4.2 1.2
```

```
#structures of data sets.
str(data_1)
```

```
## 'data.frame':    27 obs. of  1 variable:
##  $ V1: chr  "Variable names are AGE, FAT, GENDER respectively," "GENDER is a Fa
```

```
str(data_2)
```

```
## 'data.frame':    20 obs. of  4 variables:
##  $ ID   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MT1  : chr  "71,27" "58,86" "74,02" "93,38" ...
##  $ MT2  : chr  "79,64" "46,61" "39,86" "49,25" ...
##  $ Final: chr  "95,49" "40,24" "66,73" "77,47" ...
```

```
str(data_3)
```

```
## 'data.frame':    45 obs. of  1 variable:
##  $ V1: chr  "5.2 2.7 3.9 1.4" "7.3 2.9 6.3 1.8" "5.7 2.6 3.5 1" "4.4 3.2 1.3 0
```

R for now sees the data_1 and data_3 values as characters, also data_2 values are seen as characters too except ID. Here I didn't convert the data set to something that can you can make analysis. That is done in part B due to analysis is asked in part B.

Question 1 - Part B

Here in this part we converted the data1.txt file into a dataframe that had AGE,FAT,GENDER variables and also GENDER variable as mentioned in the txt turned into a factor.

```
# Install and load reader R package
#install.packages("reader") # to skip some lines.
library("reader")
```

```
## Warning: package 'reader' was built under R version 4.1.2
```

```
## Loading required package: NCmisc
```

```
## Warning: package 'NCmisc' was built under R version 4.1.2
```

```
##
## Attaching package: 'reader'
```

```
## The following objects are masked from 'package:NCmisc':
##
##     cat.path, get.ext, rmv.ext
```

```
library("stringr") # For str_trim

data_1_formatted =
  n.readLines("E:/2021-2022 Fall - Spring Books/STAT291/data1.txt",
                                n = nrow(data_1)-1,
                                skip = 2)

data_1_formatted =
  as.data.frame(do.call(rbind,
                        strsplit(data_1_formatted, split=" {2,10}")),
                                stringsAsFactors=FALSE)

names(data_1_formatted) = c("AGE","FAT","GENDER")

data_1_formatted$AGE = as.integer(data_1_formatted$AGE)
data_1_formatted$FAT = as.double(data_1_formatted$FAT)
data_1_formatted$GENDER = as.factor(data_1_formatted$GENDER)

summary(data_1_formatted$AGE[which(data_1_formatted$GENDER == "m")])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   23.00   27.00   34.00   35.53   43.00   60.00
```

```
summary(data_1_formatted$AGE[which(data_1_formatted$GENDER == "f")])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   26.00   28.00   34.00   38.20   50.75   57.00
```

Here we can see by the results that minimum age of males are lower than females, also maximum age in the group for males is also bigger. On the other hand male mean is lower than the female mean or one can say 2nd Quartile. Also one can see that male data is more compact, with some outliers, but female counterpart of the data set is more sparse.

Question 1 - Part C

Here we will order the data1 by Age and Gender then print the first 10 rows of the new ordered data frame.

```
ordered_data_1 <- data_1_formatted[with(data_1_formatted,
                                        order(data_1_formatted[,1],
                                              data_1_formatted[,3])),]
# [,3] is Gender, [,1] is AGE so ordered by both columns.

head(ordered_data_1, n = 10)
```

```
##    AGE  FAT GENDER
## 1   23 19.2      m
## 5   23 34.5      m
## 12  26 19.9      f
## 22  26 19.9      m
## 19  27 34.6      f
## 6   27 42.0      m
## 16  27 32.0      m
## 2   28 16.6      f
## 21  28 34.6      f
## 10  28 19.9      m
```

Here it is seen the first 10 rows of the ordered data set of the data1 by Age and Gender together. Since gender is factor both f and m have integer values and it is ordered by that.

Question 1 - Part D

```
data_3_formatted = read.table("E:/2021-2022 Fall - Spring Books/STAT291/data3.txt"
                              header= FALSE,
                              stringsAsFactors = FALSE)

data_3_formatted[duplicated(data_3_formatted),]
```

```
##     V1  V2  V3  V4
## 9  4.4 3.2 1.3 0.2
## 14   6 2.2   5 1.5
## 20 5.9 3.2 4.8 1.8
## 21 4.9 3.6 1.4 0.1
## 26 6.1 2.8   4 1.3
## 30   5 3.2 1.2 0.2
## 31 4.9 3.6 1.4 0.1
## 38   5 3.2 1.2 0.2
## 39 5.5 2.4 3.7   1
## 45 6.3 3.3   6 2.5
```

```
nrow(data_3_formatted[duplicated(data_3_formatted),])/nrow(data_3_formatted)
```

```
## [1] 0.2222222
```

```
for(i in 1:nrow(data_3_formatted)){
  for(i1 in 1:4){
    if(data_3_formatted[i,i1] == "**"){
      data_3_formatted[i,i1] = NA
    }
  }
}


data_3_formatted =
  na.omit(data_3_formatted[!duplicated(data_3_formatted),])

tail(data_3_formatted, n = 10)
```

```
##     V1  V2  V3  V4
## 29   5   2 3.5   1
## 32 5.6 2.5 3.9 1.1
## 33 7.1   3 5.9 2.1
## 34 6.4 3.2 4.5 1.5
## 36 5.6   3 4.1 1.3
```

```
## 37 5.5 2.4 3.7   1
## 40 5.7 2.8 4.5 1.3
## 42 7.7 3.8 6.7 2.2
## 43   6 2.2   4   1
## 44 6.2 2.9 4.3 1.3
```

Here at the end we omitted na rows and duplicated values and override our data frame and saved these into it.

Question 1 - Part E

Here it is asked from us that to write a conversion code from data frame in R to csv file.

```
write.csv(data_3_formatted,"E:/2021-2022 Fall - Spring Books/STAT291/BurkanBereket
          row.names = FALSE)
```

By this it is the end of Question one, and file is saved.

Question 2

In the question description it is given that a course consists of 20 participants, and there are 2 midterms and 1 final exam, in which all are in the data2.csv . Scoring of the course is given as Score = 0.3 * MT1 + 0.3 * MT2 + 0.4 * Final. It is asked us that to run the codeblock given.

```
library("plyr")
```

```
## Warning: package 'plyr' was built under R version 4.1.2
```

```
data_2$MT1 = as.double(gsub(",",".",gsub("\\.","",data_2$MT1)))
data_2$MT2 = as.double(gsub(",",".",gsub("\\.","",data_2$MT2)))
data_2$Final = as.double(gsub(",",".",gsub("\\.","",data_2$Final)))


data_2 <- rbind(data_2,
sample(50:100,size=3),
sample(30:90,size=3))

data_2$ID[21] = 21
data_2$ID[22] = 22

Grade_letter = c()

for(i in 1:nrow(data_2)){
  Grade_letter =
    c(Grade_letter, data_2[i,2] * 0.3 + data_2[i,3] * 0.3 + data_2[i,4]* 0.4)
}
```

```r
Letter_Grade = c()

for(i in 1:length(Grade_letter)){
  if(as.integer(Grade_letter[i])>= 90){
    Letter_Grade = c(Letter_Grade, "AA")
  } else if(as.integer(Grade_letter[i]) <= 89 && as.integer(Grade_letter[i]) >= 85
    Letter_Grade = c(Letter_Grade, "BA")
  } else if(as.integer(Grade_letter[i]) <= 84 && as.integer(Grade_letter[i]) >= 80
    Letter_Grade = c(Letter_Grade, "BB")
  } else if(as.integer(Grade_letter[i]) <= 79 && as.integer(Grade_letter[i]) >= 75
    Letter_Grade = c(Letter_Grade, "CB")
  } else if(as.integer(Grade_letter[i]) <= 74 && as.integer(Grade_letter[i]) >= 70
    Letter_Grade = c(Letter_Grade, "CC")
  } else if(as.integer(Grade_letter[i]) <= 69 && as.integer(Grade_letter[i]) >= 65
    Letter_Grade = c(Letter_Grade, "DC")
  } else if(as.integer(Grade_letter[i]) <= 64 && as.integer(Grade_letter[i]) >= 60
    Letter_Grade = c(Letter_Grade, "DD")
  } else if(as.integer(Grade_letter[i]) <= 59 && as.integer(Grade_letter[i]) >= 50
    Letter_Grade = c(Letter_Grade, "FD")
  } else{
    Letter_Grade = c(Letter_Grade, "FF")
  }
}

scores_of_students = c()

Grade_letter = cbind(Grade_letter,Letter_Grade)
scores_of_students = cbind(data_2$ID,Grade_letter)
scores_of_students = as.data.frame(scores_of_students)
names(scores_of_students) = c("ID", "Grade Average", "Letter Grade")


scores_of_students$`Grade Average` = as.double(scores_of_students$`Grade Average`)
scores_of_students$ID = as.integer(scores_of_students$ID)
scores_of_students
```

```
##    ID Grade Average Letter Grade
```

```
## 1   1        83.469           BB
## 2   2        47.737           FF
## 3   3        60.856           DD
## 4   4        73.777           CC
## 5   5        64.811           DD
## 6   6        70.471           CC
## 7   7        66.802           DC
## 8   8        73.246           CC
## 9   9        63.720           DD
## 10 10        63.002           DD
## 11 11        65.127           DC
## 12 12        68.375           DC
## 13 13        63.767           DD
## 14 14        60.768           DD
## 15 15        76.423           CB
## 16 16        65.594           DC
## 17 17        77.662           CB
## 18 18        81.758           BB
## 19 19        90.129           AA
## 20 20        67.422           DC
## 21 21        80.600           BB
## 22 22        62.400           DD
```

```
scores_of_students$`Letter Grade` = as.factor(scores_of_students$`Letter Grade`)

count(scores_of_students$`Letter Grade`)
```

```
##    x freq
## 1 AA    1
## 2 BB    3
## 3 CB    2
## 4 CC    3
## 5 DC    5
## 6 DD    7
## 7 FF    1
```

Here in this question we made a data frame from average grades of the students in the course, and then assigned them letter grades by comparing their grade with

the intervals, when it returned true for that value we gave that letter grade. Later we assigned Id's that they priorly had. Checked the frequencies of how many of each letter grade assigned to sum of students.

Question 3

Here in this question, I needed to write a function that will give me the Collatz conjecture for a given positive integer. Definition of the function is given in the description the homework.

```
Collatz <- function(x){
  a <- double()
  while(TRUE){
    if(x == round(x) && x >= 1){
      a = c(a,x)
      if(x %% 2 == 0){
      x = x/2
      } else{
          if(x == 1){
            return(a)
            break
        } else{
        x = 3*x + 1
        }
      }
    } else{
      stop('x is not a positive integer.')
    }
  }
}

Collatz(11)
```

```
##  [1] 11 34 17 52 26 13 40 20 10  5 16  8  4  2  1
```

```
Collatz(27)
```

```
##   [1]   27   82   41  124   62   31   94   47  142   71  214  107  322  161  48
##  [16]  242  121  364  182   91  274  137  412  206  103  310  155  466  233  70
##  [31]  350  175  526  263  790  395 1186  593 1780  890  445 1336  668  334  16
##  [46]  502  251  754  377 1132  566  283  850  425 1276  638  319  958  479 143
##  [61]  719 2158 1079 3238 1619 4858 2429 7288 3644 1822  911 2734 1367 4102 205
##  [76] 6154 3077 9232 4616 2308 1154  577 1732  866  433 1300  650  325  976  48
##  [91]  244  122   61  184   92   46   23   70   35  106   53  160   80   40   2
## [106]   10    5   16    8    4    2    1
```

```
Collatz(-5)
```

```
## Error in Collatz(-5): x is not a positive integer.
```

```
Collatz(6.5)
```

```
## Error in Collatz(6.5): x is not a positive integer.
```

We made a function, that does the function mentioned above. The output is the values that at each step the variable we first put took. I couldn't give the values directly side by side so I used cat function to do that.

Question 4

Here in this question it is asked from us that we need to write a function that can check whether a positive integer is prime or not, it also should check whether a given value is integer or even positive or not. It can also be both not integer and positive hence we also need to check that one out.

```
prime_func <- function(x){
  if(x >= 1){
    if(x == round(x)){
      if(any(x %% 2:(sqrt(x)) == 0)){
        paste(x,"is not a Prime Number.")
      } else{
        paste(x,"is a Prime Number")
      }
    } else{
      paste(x,"is not an integer !!!")
    }
  } else{
    if(x == round(x)){
      paste(x,"is not a postive integer !!!")
    } else{
      paste(x,"is not positive, and not integer !!!")
    }
  }
}

for (x in c(5, 4, 2.5, -4, 123, 1907)){
  print(prime_func(x))
}
```

```
## [1] "5 is a Prime Number"
## [1] "4 is not a Prime Number."
## [1] "2.5 is not an integer !!!"
## [1] "-4 is not a postive integer !!!"
## [1] "123 is not a Prime Number."
## [1] "1907 is a Prime Number"
```

In the given function we checked all the things that are asked from us, which are given in the question description.

Question 5

Here in this question it is asked from us that

```
number_to_text <- function(x){
  if(x >= 0 && x <= 999){
    single_digits <- list(zero=0, one=1, two=2, three=3, four=4, five=5,
                          six=6, seven=7, eight=8, nine=9)
    teens <- list(eleven=11, twelve=12, thirteen=13, fourteen=14, fifteen=15,
                  sixteen=16, seventeen=17, eighteen=18, nineteen=19)
    two_digits_w_o_teens <- list(ten=10, twenty=20, thirty=30, forty=40, fifty=50
                          sixty=60, seventy=70, eighty=80, ninety=90)
    two_digits <- c(teens,two_digits_w_o_teens)
    if(x >= 100 && x <= 999){

      combination = paste(names(two_digits[which(two_digits == ((x - x%%10)/10 -
                          names(single_digits[which(single_digits == x %% 10)]),
                          sep = "-")

      paste(names(single_digits[which(single_digits == (x - x %% 100)/100)]),
            "hundred and",
            combination)

    } else if(x >= 10 && x <= 99){
      if(x >= 10 && x <= 19){
        paste(names(two_digits[which(two_digits == x %% 20)])) # anything above 1

      } else{

        combination = paste(names(two_digits[which(two_digits == ((x - x%%10)/10 -
```

```
                              names(single_digits[which(single_digits == x %% 10)])
                              sep = "-")
        paste(combination)
      }

    } else{
      paste(names(single_digits[which(single_digits == x %% 10)]))
    }
  } else{
    paste("x is not in the interval [0,999]")
  }
}

for(x in c(0,5,10,15,57,538,999,-22)){
  print(number_to_text(x))
}
```

```
## [1] "zero"
## [1] "five"
## [1] "ten"
## [1] "fifteen"
## [1] "fifty-seven"
## [1] "five hundred and thirty-eight"
## [1] "nine hundred and ninety-nine"
## [1] "x is not in the interval [0,999]"
```