Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

Abstract

Data preparation is an integral part in the data analytics process. In this paper, various data cleansing techniques were performed specifically on the female runner data from the Cherry Blossom Ten Mile Run from the years 1999 through 2012. Several built-in R functions and custom functions were used to align the data for statistical analysis. The summary statistics were calculated on the cleansed data to determine the average age and run times of all female runners throughout the 14 years of races.

Introduction

The data source contains the ages and race times of participants in the Cherry Blossom Ten Mile Run held in Washington D.C. each April since 1973. In this Case Study, the data was analyzed from the years 1999 through 2012. The goal was to provide standard summary statistics for the female runners' age and run time. Before statistical analysis could be performed, several manipulations of the dataset were needed.

Methods

The data were collected from the Credit Union Cherry Blossom Ten Mile Run & 5K Run-Walk website (http://www.cherryblossom.org/). 14 years of data (1999 – 2012) were scraped from the website. For each year, the place, runner number, name, age, hometown, net time and official time were collected on each runner and stored in a text file. All of these text files needed to be cleaned in order to perform any type of meaningful analysis. Not all years were formatted the same, and columns not identical across the years. Extraction of the variables from each file was needed. This was done using the function, *extractVariables()*.

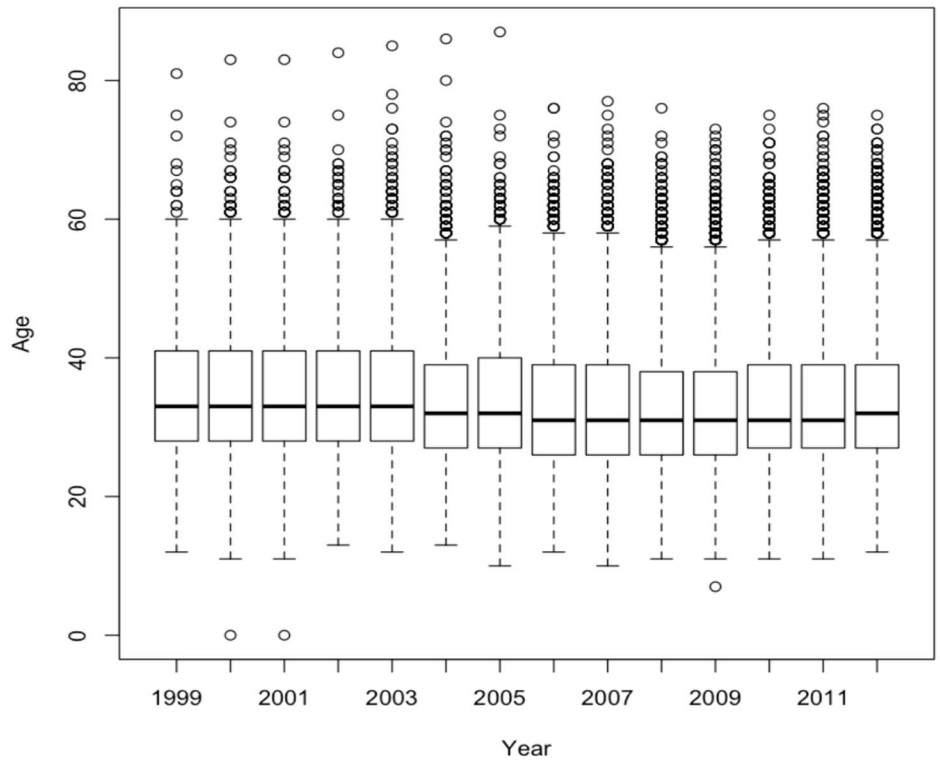
Two functions were used before *extractVariables().* The first function, *findColLocs()*, finds both the starting and ending positions of the columns. Using the function *gregexpr()*, the spaces between characters is identified. Then using R's *substring()* function, an *if* statement is written with the following logic: if a string is not equal to space, " ", then return the space plus 1, else return the space. This *if* statement will be able to differentiate a string from space and thus will be able to find the columns in our data. The second function, *selectCols()*, is created to select the columns with the data. This function takes in three arguments: the names of the columns, the header row that contains the names of the columns and the locations of the spaces. The function *sapply()*, goes through each column name and grabs the starting position of the column name and then indexes it with the given columns' name. These two functions make the code modular and make it easier to follow and be modified.

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

The main function, *extractVariables()*, takes two arguments: a file name and column names. It performs three major tasks for cleaning the data. First, each file starts with a description of what the file is and ends with a line of equal signs (=). The data of interest is after the equal signs. The index of the rows is then found with the equal signs the function *grep()* is used to discard the equal sign and everything preceding it. Next, extraction of the the two key rows and data was needed. Finally, using ther two functions described in the previous paragraph, the column names in the data are found then selected and indexed with the given input name. Figure 1 displays the distribution of ages for each race for each year.

Figure 1. Original side-by-side boxplots of age by race year



Now the data is converted into a data frame. The function, *createDF()*,converts the matrix into a data frame takes three arguments: res, year and sex. It determines which time variable to use out of the three given in the data based on the NA values each has. The function also converts the time variables into a readable time, to understand it better when performing the analysis, *convertTime()*. Each variable separates the time by a colon, ":". Using the function *strsplit()*, each instance of time is split by the colon and then converted into numeric data types. Once the time is chosen and converted, the function *createDF()*, is used to create the data frame using *data.frame()*.

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

Results

After the raw data was imported into R, cleaned and formatted, investigation of the data frame is needed to ensure the data is ready for analysis. This data frame has 6 attributes – *year*, *sex*, *name*, *home*, *age* and *runtime* – as shown in Figure 3. We have 3 numeric variables in this data – *year*, *age* and *runtime* – whereas the other 3 – *sex*, *name* and *home* – are categorical. The total number of records, 74, 237, represents the total female participation in the Cherry Blossom race between 1999 and 2012.

Figure 2. Data-frame for female runners

```
'data.frame':   74237 obs. of  6 variables:
 $ year   : int  1999 1999 1999 1999 1999 1999 1999 1999 1999 1999 ...
 $ sex    : chr  "W" "W" "W" "W" ...
 $ name   : chr  "Jane Omoro            ", "Jane Ngotho           " "Lidiya Grigoryeva     " "Eunice Sagero         " ...
 $ home   : chr  "Kenya              " "Kenya              " "Russia             " "Kenya              " ...
 $ age    : num  26 29 NA 20 29 24 38 NA 27 30 ...
 $ runTime: num  53.6 53.6 53.7 53.9 54.1 ...
```

A sample of records are displayed in Table 1.

Table 1. Sample records

	year	sex	name	home	age	runTime
1999.1	1999	W	Jane Omoro	Kenya	26	53.61667
1999.2	1999	W	Jane Ngotho	Kenya	29	53.63333
1999.3	1999	W	Lidiya Grigoryeva	Russia	NA	53.66667
1999.4	1999	W	Eunice Sagero	Kenya	20	53.91667
1999.5	1999	W	Alla Zhilyayeva	Russia	29	54.13333
1999.6	1999	W	Teresa Wanjiku	Kenya	24	54.16667

The summary statistics for all the variables are shown in Table 2 and Table 3. These summary statistics confirm the presence of all 14 years (1999 - 2012) in the data. The figure also shows that the length of all of the character variables 74, 237. The attributes – *age* and *runtime* – have 24 and 5,434 missing observations, respectively. The oldest female who participated in the Cherry Blossom race was 89 years of age. The average age of female participants is 34 years old and 75% are younger than 40 years.

The finish-time of the womens' ten-mile race was wide ranging. The average finish-time was 96.93 minutes, whereas 25% of the participants finished it below 87.25 minutes and 75% completed in less than 105.97 minutes. The fastest time recorded in the female ten-mile run over 14 years is 45.25 minutes while the slowest time is 177.52 minutes.

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

Table 2. Summary statistics for character variables for female runners

Statistic	Year	Age	Runtime
Length	74,237	74,237	74,237

Table 3. Summary statistics for numeric variables for female runners

Statistic	Year	Age	Runtime
Minimum	1999	0.00	45.25
1st Quartile	2005	27.00	87.25
Median	2008	32.00	96.35
Mean	2007	34.21	96.93
3rd Quartile	2010	40.00	105.97
Maximum	2012	89.00	177.52
NA's		25	5,434

Possible Future Work

With the data in the correct format, further analysis can be performed. One possibility for analysis would involve fitting models to average the performance of runners. This model could be used to predict the run time based on age.

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

References

1. Deborah Nolan and Duncan Temple Lang, “Case Studies in Data Science with R”. University of California, Berkeley and University of California, Davis. 2015, pp. 45-63. http://www.rdatasciencecases.org

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

Appendix – R Code

```
femalefilenames = paste("C:/Users/Gabri/OneDrive/Documents/SMU/Data/DS7333/",
1999:2012, ".txt", sep = "")
femaleFiles = lapply(femalefilenames, readLines)

names(femaleFiles) = 1999:2012

findColLocs = function(spacerRow) {
  spaceLocs = gregexpr(" ", spacerRow)[[1]]
  rowLength = nchar(spacerRow)
  if (substring(spacerRow, rowLength, rowLength) != " ")
    return( c(0, spaceLocs, rowLength + 1))
  else return(c(0, spaceLocs))
}

selectCols = function(shortColNames, headerRow, searchLocs) {
  sapply(shortColNames, function(shortName, headerRow, searchLocs){
    startPos = regexpr(shortName, headerRow)[[1]]
    if (startPos == -1) return( c(NA, NA) )
    index = sum(startPos >= searchLocs)
    c(searchLocs[index] + 1, searchLocs[index + 1])
  }, headerRow = headerRow, searchLocs = searchLocs )
}

extractVariables =
  function(file, varNames =c("name", "home", "ag", "gun",
                             "net", "time"))
{
      # Find the index of the row with =s
  eqIndex = grep("^===", file)
      # Extract the two key rows and the data
  spacerRow = file[eqIndex]
  headerRow = tolower(file[ eqIndex - 1 ])
  body = file[ -(1 : eqIndex) ]

      # Obtain the starting and ending positions of variables
  searchLocs = findColLocs(spacerRow)
  locCols = selectCols(varNames, headerRow, searchLocs)

  Values = mapply(substr, list(body), start = locCols[1, ],
                  stop = locCols[2, ])
  colnames(Values) = varNames
```

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

```
  invisible(Values)
}

femaleResMat = lapply(femaleFiles, extractVariables)
length(femaleResMat)

femaleFiles[['2001']][1:15]

men2001 = readLines("C:/Users/Gabri/OneDrive/Documents/SMU/Data/DS7333/men2001.txt")
men2001[1:15]

femaleFiles[['2000']][9:10] = men2001[12:13]
femaleFiles[['2000']][1:15]

femaleFiles[['2001']][9:10] = men2001[12:13]
femaleFiles[['2001']][1:15]

femaleResMat = lapply(femaleFiles, extractVariables)
length(femaleResMat)

sapply(femaleResMat, nrow)

age = as.numeric(femaleResMat[['2012']][ , 'ag'])

age = sapply(femaleResMat,
             function(x) as.numeric(x[ , 'ag']))

boxplot(age, ylab = "Age", xlab = "Year")

selectCols = function(shortColNames, headerRow, searchLocs) {
  sapply(shortColNames, function(shortName, headerRow, searchLocs){
    startPos = regexpr(shortName, headerRow)[[1]]
    if (startPos == -1) return( c(NA, NA) )
    index = sum(startPos >= searchLocs)
    c(searchLocs[index] + 1, searchLocs[index + 1])
  }, headerRow = headerRow, searchLocs = searchLocs )
}

femaleResMat = lapply(femaleFiles, extractVariables)
age = sapply(femaleResMat,
             function(x) as.numeric(x[ , 'ag']))
```

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

```
boxplot(age, ylab = "Age", xlab = "Year")

sapply(age,  function(x) sum(is.na(x)))
convertTime = function(time) {
  timePieces = strsplit(time, ":")
  timePieces = sapply(timePieces, as.numeric)
  sapply(timePieces, function(x) {
                    if (length(x) == 2) x[1] + x[2]/60
                    else 60*x[1] + x[2] + x[3]/60
                    })
}

createDF =
function(Res, year, sex)
{
      # Determine which time to use
  useTime = if( !is.na(Res[1, "net"]) )
              Res[ , "net"]
            else if( !is.na(Res[1, "gun"]) )
              Res[ , "gun"]
            else
              Res[ , "time"]
  runTime = convertTime(useTime)
  Results = data.frame(year = rep(year, nrow(Res)),
                       sex = rep(sex, nrow(Res)),
                       name = Res[ , "name"],
                       home = Res[ , "home"],
                       age = as.numeric(Res[, "ag"]),
                       runTime = runTime,
                       stringsAsFactors = FALSE)
  invisible(Results)
}

femaleDF = mapply(createDF, femaleResMat, year = 1999:2012,
                  sex = rep("W", 14), SIMPLIFY = FALSE)

sapply(femaleDF, function(x) sum(is.na(x$runTime)))


createDF = function(Res, year, sex)
{
  # Determine which time to use
  if ( !is.na(Res[1, 'net']) ) useTime = Res[ , 'net']
```

Case Study Unit 8: Cherry Blossom Race Data

Gabriel Gonzales
DS 7333 - QTW
9/23/2020

```
  else if ( !is.na(Res[1, 'gun']) ) useTime = Res[ , 'gun']
  else useTime = Res[ , 'time']

  # Remove # and * and blanks from time
  useTime = gsub("[#\\*[:blank:]]", "", useTime)
  runTime = convertTime(useTime[ useTime != "" ])

  # Drop rows with no time
  Res = Res[ useTime != "", ]

  Results = data.frame(year = rep(year, nrow(Res)),
                       sex = rep(sex, nrow(Res)),
                       name = Res[ , 'name'], home = Res[ , 'home'],
                       age = as.numeric(Res[, 'ag']),
                       runTime = runTime,
                       stringsAsFactors = FALSE)
  invisible(Results)
}

femaleDF = mapply(createDF, femaleResMat, year = 1999:2012,
                  sex = rep("W", 14), SIMPLIFY = FALSE)

sapply(femaleDF, function(x) sum(is.na(x$runTime)))

cbFemale = do.call(rbind, femaleDF)
save(cbFemale, file = "cbFemale.rda")

dim(cbFemale)

head(cbFemale)

str(cbFemale)

summary(cbFemale)
```