Olympic Data Study

Dan Burke, Rajinder Singh, Marley Akonnor

R Markdown

```
#Created a function to Check for NA values
checkForNaValuesInColumn <- function(vec, colname){
  numberOfNas <-length(vec[is.na(vec)])
  print(colname)
  print(numberOfNas)
}

#read the CSV, Create a Data Frame from each Dataset
athleteEvents <- data.frame(read.csv("C:\\Users\\danbu\\Desktop\\OlympicData\\athlete_events.csv"))
nocRegions <- data.frame(read.csv("C:\\Users\\danbu\\Desktop\\OlympicData\\noc_regions.csv"))

#Grab the column names for reference
athleteColumnNames <- colnames(athleteEvents)
regionColnames <- colnames(nocRegions)

#Replace NA values for Medals with "None", we can assume a NA == no medal, however we can't assume the
athleteEvents$Medal[is.na(athleteEvents$Medal)] <- "None"

#check For NA values
checkForNaValuesInColumn(athleteEvents$Name, "Name")
```

```
## [1] "Name"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Sex, "Sex")
```

```
## [1] "Sex"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Age, "Age")
```

```
## [1] "Age"
## [1] 9474
```

```
checkForNaValuesInColumn(athleteEvents$Height, "Height")
```

```
## [1] "Height"
## [1] 60171
```

```
checkForNaValuesInColumn(athleteEvents$Weight, "Weight")
```

```
## [1] "Weight"
## [1] 62875
```

```
checkForNaValuesInColumn(athleteEvents$Team, "Team")
```

```
## [1] "Team"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$NOC, "NOC")
```

```
## [1] "NOC"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Games, "Games")
```

```
## [1] "Games"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Year, "Year")
```

```
## [1] "Year"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Season, "Season")
```

```
## [1] "Season"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Sport, "Sport")
```

```
## [1] "Sport"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$City, "City")
```

```
## [1] "City"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Event, "Event")
```

```
## [1] "Event"
## [1] 0
```

```
checkForNaValuesInColumn(athleteEvents$Medal, "Medal")
```

```
## [1] "Medal"
## [1] 0
```

```
#Omit all NA rows within the Athlete Events Data Frame

athletes1 <- na.omit(athleteEvents)



print("Number of Omitted Rows")
```

```
## [1] "Number of Omitted Rows"
```

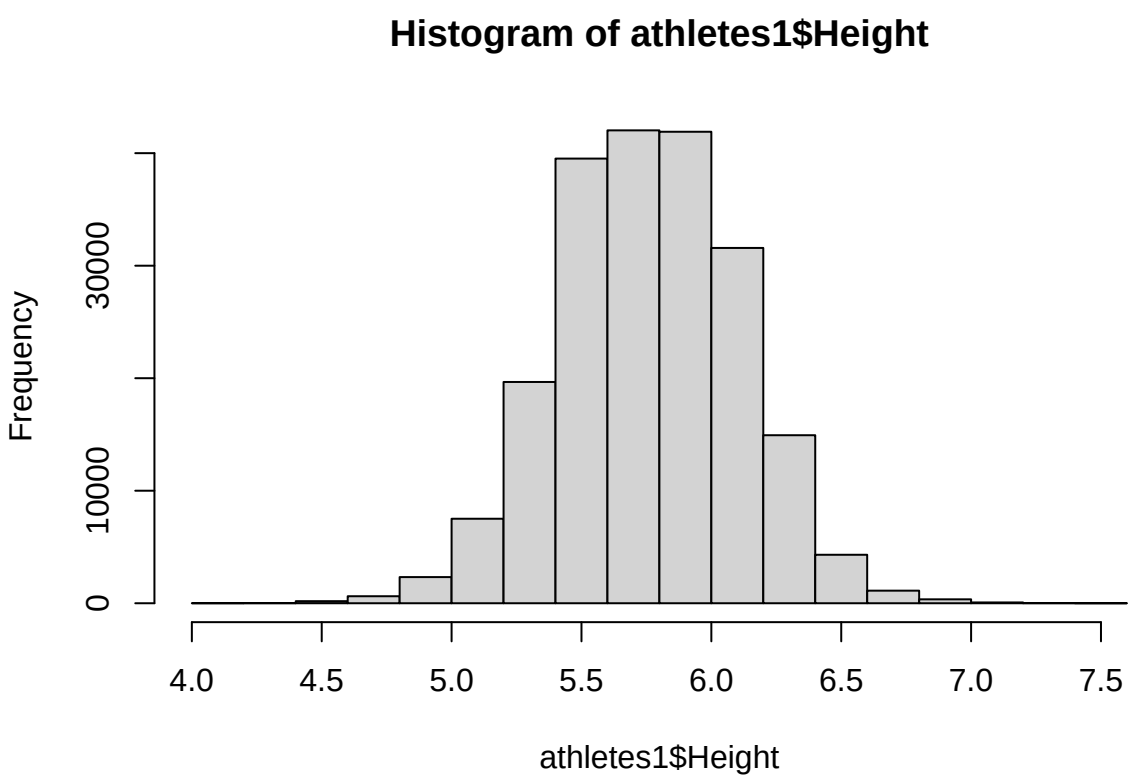
```
print(length(athleteEvents$ID)-length(athletes1$ID))
```

```
## [1] 64951
```

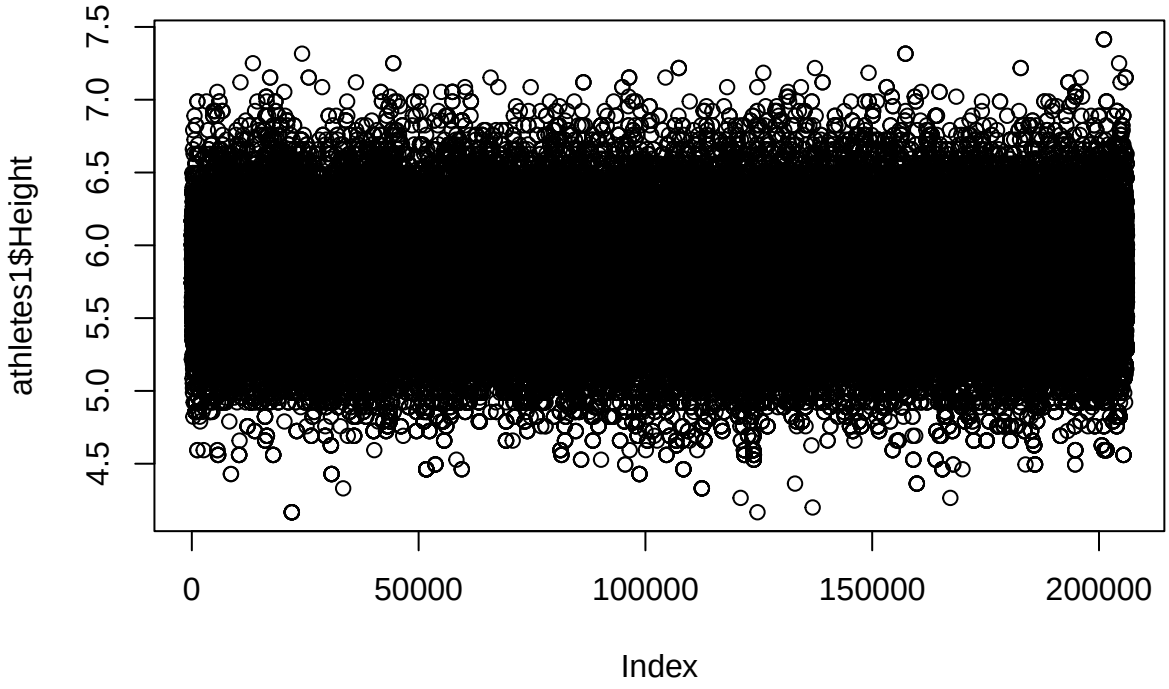
```
#Reivew the Data Frame
str(athletes1)
```

```
## 'data.frame':    206165 obs. of  15 variables:
##  $ ID    : int  1 2 5 5 5 5 5 5 6 6 ...
##  $ Name  : chr  "A Dijiang" "A Lamusi" "Christine Jacoba Aaftink" "Christine Jacoba Aaftink" ...
##  $ Sex   : chr  "M" "M" "F" "F" ...
##  $ Age   : int  24 23 21 21 25 25 27 27 31 31 ...
##  $ Height: int  180 170 185 185 185 185 185 185 188 188 ...
##  $ Weight: num  80 60 82 82 82 82 82 82 75 75 ...
##  $ Team  : chr  "China" "China" "Netherlands" "Netherlands" ...
##  $ NOC   : chr  "CHN" "CHN" "NED" "NED" ...
##  $ Games : chr  "1992 Summer" "2012 Summer" "1988 Winter" "1988 Winter" ...
##  $ Year  : int  1992 2012 1988 1988 1992 1992 1994 1994 1992 1992 ...
##  $ Season: chr  "Summer" "Summer" "Winter" "Winter" ...
##  $ City  : chr  "Barcelona" "London" "Calgary" "Calgary" ...
##  $ Sport : chr  "Basketball" "Judo" "Speed Skating" "Speed Skating" ...
##  $ Event : chr  "Basketball Men's Basketball" "Judo Men's Extra-Lightweight" "Speed Skating Women's 5
##  $ Medal : chr  "None" "None" "None" "None" ...
##  - attr(*, "na.action")= 'omit' Named int [1:64951] 3 4 27 28 30 36 37 38 39 40 ...
##   ..- attr(*, "names")= chr [1:64951] "3" "4" "27" "28" ...
```

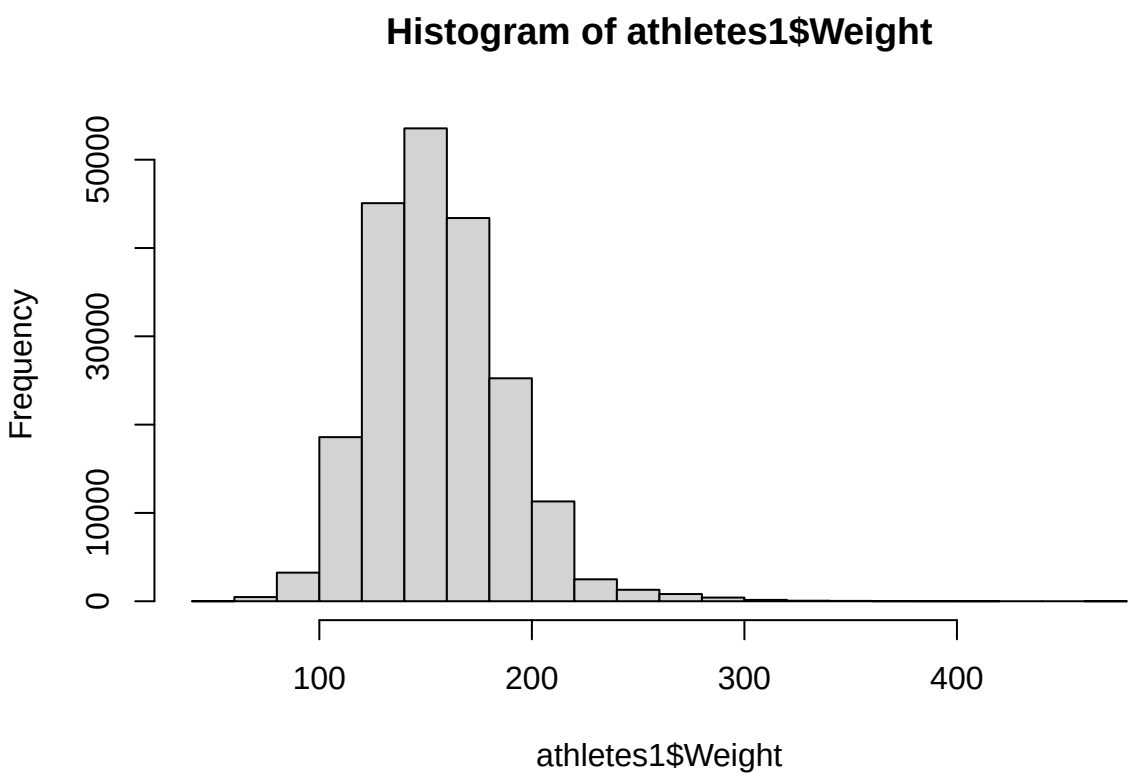
```
#Convert CM to Feet (Decimal)
athletes1$Height <- athletes1$Height / 30.48

#Convert KG to lbs

athletes1$Weight <- athletes1$Weight * 2.2

#View Distribution
hist(athletes1$Height)
```



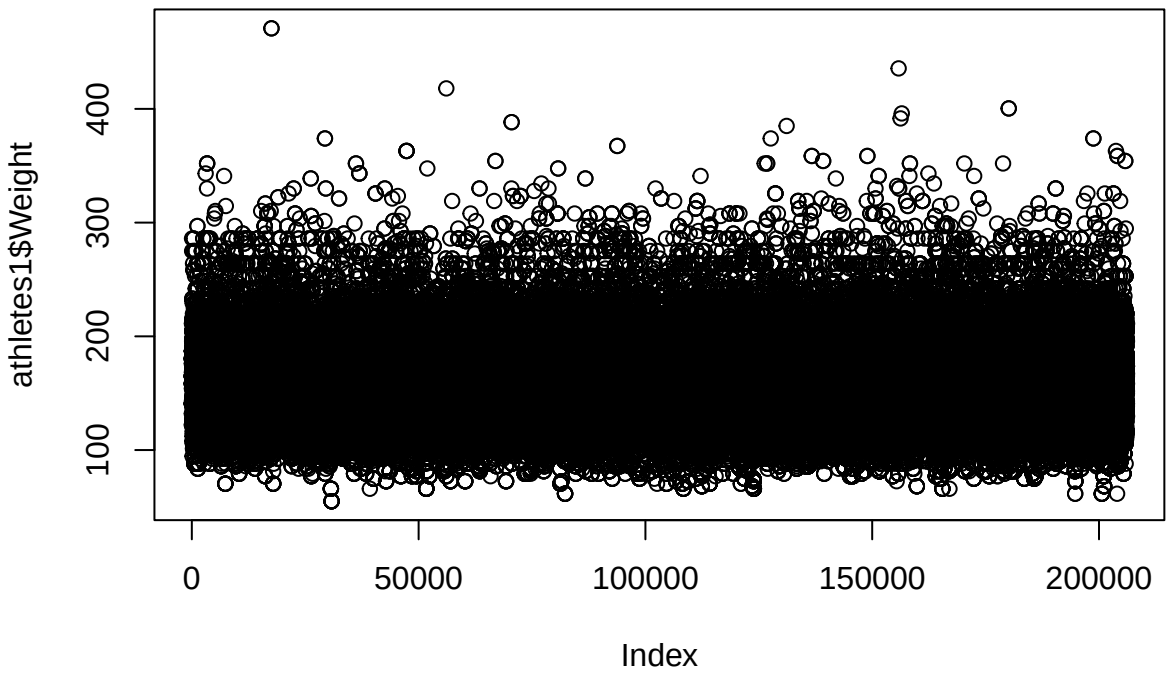

```
plot(athletes1$Height)
```




```
hist(athletes1$Weight)
```




```
plot(athletes1$Weight)
```




```
summary(athletes1$Height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.167   5.512   5.741   5.754   6.004   7.415
```

```
summary(athletes1$Weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    55.0   132.0   154.0   155.5   173.8   470.8
```

```
countries <- read.csv("C:\\Users\\danbu\\Desktop\\OlympicData\\allCountries.csv")

#Clean up Country Name Strings
library(stringr)

#Match up Countries & Athletes we have locations for, with the change in country names this will affect
matchedCoAthIndex<- athletes1[match(athletes1$Team, countries$name),]
matchedCoAthIndex <- na.omit(matchedCoAthIndex)

#Get the Athletes who have never medal'd,We know that countries can medal and not medal one year so we
#must find countries which have not recieved a medal at all
countries <- unique(athletes1$Team)
countries <- lapply(countries, sub, pattern = '\\-1', replacement = '')
countries <- lapply(countries, sub, pattern = '\\-2', replacement = '')
```

```
countries <- lapply(countries, sub, pattern = '\\-3', replacement = '')
countryMedalSum <- data.frame()

for(i in 1:length(countries)){
  countryMedalSum[i,1] <- countries[i]
  countryMedalSum[i,2] <- nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Gold
  countryMedalSum[i,3] <- nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Silv
  countryMedalSum[i,4] <- nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Bron
  countryMedalSum[i,5] <- nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="None
  #Get and Add the sum for each medal
  countryMedalSum[i,6] <- sum(
    nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Gold"),]),
    nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Silver"),]),
    nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="Bronze"),]),
    nrow(athletes1[which(athletes1$Team == countries[i] & athletes1$Medal =="None"),])
  )
}



colnames(countryMedalSum) <- c("team", "Gold", "Silver", "Bronze", "None", "Total Athletes")

countryGeo <- read.csv("C:\\Users\\danbu\\Desktop\\OlympicData\\allCountries.csv")
countryMedalSum <- countryMedalSum[match(countryGeo$name,countryMedalSum$team),]
countryMedalSum <-na.omit(countryMedalSum)

for(i in 1:length(countryMedalSum$team)){
  countryMedalSum$lats[i] <- countryGeo[which(countryGeo$name == countryMedalSum[i,1]), 2]
  countryMedalSum$longs[i] <- countryGeo[which(countryGeo$name == countryMedalSum[i,1]), 3]
}

colnames(countryMedalSum) <- c("team", "Gold", "Silver", "Bronze", "None", "Total Athletes", "lat", "lo

#Write it to a csv in order to keep the siny app more simple

path_out = "C:\\Users\\danbu\\Documents\\GitHub\\IST687\\Dan\\Data Cleaning Etc\\"
fileName = paste(path_out, 'medalSums.csv',sep = '')
write.csv(countryMedalSum,fileName)



library(shiny)
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.1.1
```

```
library(shinydashboard)
```

```
## Warning: package 'shinydashboard' was built under R version 4.1.1
```

```
##
## Attaching package: 'shinydashboard'
```

```
## The following object is masked from 'package:graphics':
##
##     box
```

```
library(scales)
library(leaflet.minicharts)
```

```
## Warning: package 'leaflet.minicharts' was built under R version 4.1.1
```

```
#install.packages("leaflet.minicharts")
#https://rstudio.github.io/leaflet/markers.html#circle-markers
countryMedals <- read.csv("C:\\Users\\danbu\\Documents\\GitHub\\IST687\\Dan\\OlympicGamesMap\\OlyMap\\m
```

```
colors <- c("#FFFF33", "#CECCB7", "#EE9A25")
MiniBarDf <- data.frame(countryMedals$Gold, countryMedals$Silver, countryMedals$Bronze)
colnames(MiniBarDf) <- c("Gold", "Silver", "Bronze")
MiniBarDf <- MiniBarDf
scaledTotalAthletes <- rescale(countryMedals$Total.Athletes, to = c(100000,750000))
```

```
ui <- dashboardPage(
    dashboardHeader(title = "IST 687 - Olympics"),
    dashboardSidebar(
        sidebarMenu(
            menuItem(
                "Maps",
                tabName = "maps",
                icon = icon("globe"),
                menuSubItem("Countries", tabName = "m_country", icon = icon("map")),
                menuSubItem("Total Athletes", tabName = "m_tot_ath", icon = icon("map"))
            ),
            menuItem(
                "Charts",
                tabName = "charts",
                icon = icon("bar-chart"),
                menuSubItem("Countries", tabName = "c_Countries", icon = icon("area-chart")),
                menuSubItem("Total Athletes", tabName = "c_Total_Athletes", icon = icon("area-chart"))
            )
        )
    ),
    dashboardBody(
        tags$style(type = "text/css", "#map {height: calc(100vh - 80px) !important;}"),
        leafletOutput("map")
    )
)

server <- function(input, output, session) {

    output$map <- renderLeaflet({
        leaflet() %>%
```

```
            addTiles(group="OSM")%>%
            setView(lat = 40.459152,
                    lng =  49.426570,
                    zoom = 4
            ) %>%
            addCircles(lng=countryMedals$long,
                       lat=countryMedals$lat,
                       radius=scaledTotalAthletes,
                       popup=paste(countryMedals$team,"Total Athletes", as.character(countryMedals$Total
                       color = "#3462eb",
                       fillOpacity = 0.15,
                       stroke=FALSE
            )%>%
            addMinicharts(countryMedals$long,
                          countryMedals$lat,
                          type = "pie",
                          chartdata = MiniBarDf,
                          colorPalette = colors,
                          legend = TRUE,
                          legendPosition = "topright",
                          width = 20, height = 20)

    })
}
```