Project 1: Olmypic History Trend Analytics

Group 9 - Thomas George Thomas, Yang Liu, Pratyush Pothuneedi

10/28/2021

```
#Importing Required packages
library(tidyverse)
library(reshape2)
library(dplyr)
library(knitr)

## Uncomment below to set the working directory.
##setwd("C:/Users/Docs")
```

1. Data Acquisition

Importing the datasets

```
# Data of the athelets and countries
atheletes_df <- read.csv('athlete_events.csv', header = TRUE, sep = ',')
head(atheletes_df, 5) # structure of the dataset
```

```
##   ID                      Name Sex Age Height Weight           Team NOC
## 1  1                 A Dijiang   M  24    180     80          China CHN
## 2  2                  A Lamusi   M  23    170     60          China CHN
## 3  3      Gunnar Nielsen Aaby   M  24     NA     NA        Denmark DEN
## 4  4     Edgar Lindenau Aabye   M  34     NA     NA Denmark/Sweden DEN
## 5  5 Christine Jacoba Aaftink   F  21    185     82    Netherlands NED
##         Games Year Season      City          Sport
## 1 1992 Summer 1992 Summer Barcelona     Basketball
## 2 2012 Summer 2012 Summer    London           Judo
## 3 1920 Summer 1920 Summer Antwerpen       Football
## 4 1900 Summer 1900 Summer     Paris     Tug-Of-War
## 5 1988 Winter 1988 Winter   Calgary Speed Skating
##                               Event Medal
## 1      Basketball Men's Basketball  <NA>
## 2     Judo Men's Extra-Lightweight  <NA>
## 3          Football Men's Football  <NA>
## 4      Tug-Of-War Men's Tug-Of-War  Gold
## 5 Speed Skating Women's 500 metres  <NA>
```

```r
# Importing Data of the regions tied with the NOC code
regions_df <- read.csv('noc_regions.csv', header= TRUE, sep =',')
head(regions_df, 5)
```

```
##   NOC      region                notes
## 1 AFG Afghanistan
## 2 AHO     Curacao Netherlands Antilles
## 3 ALB     Albania
## 4 ALG     Algeria
## 5 AND     Andorra
```

2. Data Wrangling

2.1 Data Discovery

A. Summary Statistics

```r
summary(atheletes_df)
```

```
##        ID             Name               Sex                 Age
##  Min.   :     1   Length:271116      Length:271116      Min.   :10.00
##  1st Qu.: 34643   Class :character   Class :character   1st Qu.:21.00
##  Median : 68205   Mode  :character   Mode  :character   Median :24.00
##  Mean   : 68249                                        Mean   :25.56
##  3rd Qu.:102097                                        3rd Qu.:28.00
##  Max.   :135571                                        Max.   :97.00
##                                                        NA's   :9474
##      Height          Weight          Team               NOC
##  Min.   :127.0   Min.   : 25.0   Length:271116      Length:271116
##  1st Qu.:168.0   1st Qu.: 60.0   Class :character   Class :character
##  Median :175.0   Median : 70.0   Mode  :character   Mode  :character
##  Mean   :175.3   Mean   : 70.7
##  3rd Qu.:183.0   3rd Qu.: 79.0
##  Max.   :226.0   Max.   :214.0
##  NA's   :60171   NA's   :62875
##     Games                Year         Season              City
##  Length:271116      Min.   :1896   Length:271116      Length:271116
##  Class :character   1st Qu.:1960   Class :character   Class :character
##  Mode  :character   Median :1988   Mode  :character   Mode  :character
##                     Mean   :1978
##                     3rd Qu.:2002
##                     Max.   :2016
##
##     Sport              Event              Medal
##  Length:271116      Length:271116      Length:271116
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

We can see that there are NA's in the numerical fields of Age, Height, Weight which we will handle

```
summary(regions_df)
```

```
##      NOC                region             notes          
##  Length:230         Length:230         Length:230        
##  Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character
```

B. Discovering Discrete Data

```
kable(
  atheletes_df %>%
    summarise(total_records=n()),
  caption = "Total Records for Athletes Dataframe"

)
```

Table 1: Total Records for Athletes Dataframe

total_records
271116

```
kable(
  regions_df %>%
    summarise(total_records=n()),
  caption = "Total Records in Regions Dataframe"

)
```

Table 2: Total Records in Regions Dataframe

total_records
230

Looking for NA's in all the columns

```
# Store the cols with missing values
list_na <- colnames(atheletes_df)[apply(atheletes_df, 2, anyNA)]
list_na
```

```
## [1] "Age"    "Height" "Weight" "Medal"
```

We have NA's for numerical data: Age, Height & Weight and for categorical data: Medal.

```
kable(
  atheletes_df %>%
    group_by(Medal) %>%
    summarise(total_records=n())
    ,caption="Records by Medal Count"

)
```

Table 3: Records by Medal Count

Medal	total_records
Bronze	13295
Gold	13372
Silver	13116
NA	231333

There are 231333 NA's for Medals which is categorical data and we need to handle this in the cleaning part

```
#looking for NA's in regions_df
kable(
  regions_df %>%
    filter(is.na(region)) %>%
    group_by(NOC,region,notes) %>%
  summarise(Total_records=n()),
  caption="Records grouped by categories"
)
```

```
## `summarise()` has grouped output by 'NOC', 'region'. You can override using the `.groups` argument.
```

Table 4: Records grouped by categories

NOC	region	notes	Total_records
ROT	NA	Refugee Olympic Team	1
TUV	NA	Tuvalu	1
UNK	NA	Unknown	1

No NA's in region_df

2.2 Structuring

```
head(atheletes_df,5)
```

```
##   ID                    Name Sex Age Height Weight           Team NOC
## 1  1               A Dijiang   M  24    180     80          China CHN
## 2  2                A Lamusi   M  23    170     60          China CHN
```

```
## 3  3      Gunnar Nielsen Aaby   M  24     NA     NA        Denmark DEN
## 4  4     Edgar Lindenau Aabye   M  34     NA     NA Denmark/Sweden DEN
## 5  5 Christine Jacoba Aaftink   F  21    185     82    Netherlands NED
##         Games Year Season      City         Sport
## 1 1992 Summer 1992 Summer Barcelona    Basketball
## 2 2012 Summer 2012 Summer    London          Judo
## 3 1920 Summer 1920 Summer Antwerpen      Football
## 4 1900 Summer 1900 Summer     Paris    Tug-Of-War
## 5 1988 Winter 1988 Winter   Calgary Speed Skating
##                              Event Medal
## 1      Basketball Men's Basketball  <NA>
## 2     Judo Men's Extra-Lightweight  <NA>
## 3          Football Men's Football  <NA>
## 4      Tug-Of-War Men's Tug-Of-War  Gold
## 5 Speed Skating Women's 500 metres  <NA>
```

We can see that we don't need to do additional restructuring as columns like "Games" is already split and available as Year and Season

2.3 Cleaning

Handling Missing Data

We can't filter out the NA values since the columns that exhibit them are required for our analysis. We will be filling the NA values for numerical columns like Age, Height, Weight with the **median** values since we require whole numbers. The Medals are filled with **'None'** which would signify that the athletes simply didn't win any of the categories of Medals (Gold, Silver, Bronze).

We didn't filter out the NA records in Age, Height and Weight because that would mean that crucial data would be dropped leading to data skewness, we are using the Median values since we require whole numbers and to reduce the degree of skewness while maintaining data integrity.

```
atheletes_df$Medal <- atheletes_df$Medal %>%
  replace_na("None") # It is assumed tha the athlete participated in the sport but didn't win a medal
```

Replacing NA's in Medals Calculating Missing Median for the missing values for Age, Height and Weight

```
list_na <- list_na[ list_na !="Medal"]

# Calculate median for the missing values
missing_median <- apply(atheletes_df[,colnames(atheletes_df) %in% list_na],
      2, # 2 is for Columns
      median,
      na.rm =  TRUE)
missing_median
```

```
##    Age Height Weight
##     24    175     70
```

```
# Replace the missing values with median
atheletes_df <- atheletes_df %>%
   mutate(
     Age  = ifelse(is.na(Age), missing_median[1], Age),
     Height = ifelse(is.na(Height), missing_median[2], Height),
     Weight = ifelse(is.na(Weight), missing_median[3], Weight)
   )
```

```
# Replacing Na's with the respective region/notes for the NOC's
regions_df$region <- ifelse(is.na(regions_df$region), regions_df$notes, regions_df$region)
```

```
kable(
  regions_df %>%
    filter(is.na(region)) %>%
    summarise(total_records=n())
  ,caption = "Number of NA's in Region after fix"

)
```

Handling Missing data in Regions

Table 5: Number of NA's in Region after fix

total_records
0

2.4 Enriching

A. Adding Attribute region

We will join regions_df and atheletes_df based on the NOC code to get the Region for enriching the data.

```
athletes <- left_join(atheletes_df, regions_df, by="NOC")

# Replacing Region with Country to make the data more meaningful
colnames(athletes)[which(names(athletes) == "region")] <- "Region"
# Removing notes since it's not relevant to our analysis anymore
athletes <- athletes[,-17]
head(athletes,5)
```

```
##   ID                      Name Sex Age Height Weight           Team NOC
## 1  1                 A Dijiang   M  24    180     80          China CHN
## 2  2                  A Lamusi   M  23    170     60          China CHN
## 3  3       Gunnar Nielsen Aaby   M  24    175     70        Denmark DEN
## 4  4      Edgar Lindenau Aabye   M  34    175     70 Denmark/Sweden DEN
## 5  5 Christine Jacoba Aaftink   F  21    185     82    Netherlands NED
```

```
##         Games Year Season      City         Sport
## 1 1992 Summer 1992 Summer Barcelona    Basketball
## 2 2012 Summer 2012 Summer    London          Judo
## 3 1920 Summer 1920 Summer Antwerpen      Football
## 4 1900 Summer 1900 Summer     Paris    Tug-Of-War
## 5 1988 Winter 1988 Winter   Calgary Speed Skating
##                              Event Medal      Region
## 1      Basketball Men's Basketball  None       China
## 2     Judo Men's Extra-Lightweight  None       China
## 3          Football Men's Football  None     Denmark
## 4      Tug-Of-War Men's Tug-Of-War  Gold     Denmark
## 5 Speed Skating Women's 500 metres  None Netherlands
```

We don't have any other attribute to split or to create a new category since we believe that we have all the required columns for our analysis

2.5 Validating

Check for any missing values

```
# Counting the number of NA's for all the columns
colnames(athletes)[apply(athletes, 2, anyNA)]
```

```
## [1] "Region"
```

```
kable(
  athletes %>%
    select(NOC,Region)  %>%
    filter(is.na(Region)) %>%
    group_by(NOC,Region) %>%
    summarise(total_records=n())
    ,caption="Null Records check by Medal Count"

)
```

```
## `summarise()` has grouped output by 'NOC'. You can override using the `.groups` argument.
```

Table 6: Null Records check by Medal Count

NOC	Region	total_records
SGP	NA	349

For NOC SGP, there are no records in our regions_df but is present in atheltes_df, as a result we are getting NA values after the join. We will add Singapore Region to the NOC in the joined data

```
athletes$Region <- ifelse(is.na(athletes$Region) && athletes$NOC=='SGP', "Singapore", athletes$Region)


kable(
  athletes %>%
    select(Region)  %>%
    filter(is.na(Region)) %>%
    group_by(Region) %>%
    summarise(total_records=n())
    ,caption="Checking for NA records in Region after change"

)
```

Table 7: Checking for NA records in Region after change

Region	total_records

Check for Duplicates

```
sum(duplicated(athletes))
```

```
## [1] 1385
```

There 1385 duplicate records on the whole data set

```
# Removing the duplicates
athletes <- unique(athletes)
```

Checking boundary cases

```
kable (
athletes %>%
summarise(max_age=max(Age), min_age=min(Age), Average_Age=mean(Age)),
caption="Age boundary cases"

)
```

Table 8: Age boundary cases

max_age	min_age	Average_Age
97	10	25.40454

```
kable (
athletes %>%
summarise(max_height=max(Height), min_height=min(Height), Average_height=mean(Height)),
caption="Height boundary cases"

)
```

Table 9: Height boundary cases

max_height	min_height	Average_height
226	127	175.265

```
kable (
athletes %>%
summarise(max_weight=max(Weight), min_weigt=min(Weight), Average_weight=mean(Weight)),
caption="Weight boundary cases"

)
```

Table 10: Weight boundary cases

max_weight	min_weigt	Average_weight
214	25	70.5417

All our boundary cases looks reasonable and accurate.

2.6 Publishing

The data is cleaned & wrangled and made available for the team to develop business cases.