Lab 6

Autumn Li and UNI ql2280

Dec 03, 2016

In today's lab we will use data on the 2829 fastest men's and women's 100m sprint times saved as dataframes `sprint.m.csv` and `sprint.f.csv`.

1. Load the dataframe `sprint.m.csv` and save it as `sprint.m`. Append a column to the dataframe called `CityDate` that is defined by concatenating the string entries in the `City` and `Date` columns. For example, entries "Berlin" and "16.08.2009" in the `City` and `Date` columns, respectively, produce an entry of "Berlin 16.08.2009" in the `CityDate` column. We assume that every unique combination of city and date in the `CityDate` column corresponds to a unique track meet. How many unique track meets occur? How many other sprint times were recorded in the same track meet as Usain Bolt's legendary time of 9.58 seconds?

```
sprint.m <- read.csv("~/Desktop/sprint.m.csv")
sprint.m$CityDate = paste(sprint.m$City, sprint.m$Date)
length(unique(sprint.m$CityDate))
```

```
## [1] 1181
```

2. Compute a reduced version of `sprint.m` that only keeps the fastest time from each track meet. For example, of all rows that correspond to sprint times recorded at the "Berlin 16.08.2009" track meet, we will only keep Usain Bolt's row since his time of 9.58 was fastest. Hint: There are many ways to do this, `tapply()` or `split()` might be helpful. You can do this without using a loop. Call the result `sprint.m.fastest` and check that the number of rows is the same as the number of unique men's track meets. Display the first five rows.

```
find.rows = function(rows,data){
  return(rows[which.min(data$Time[rows])])
}
sprint.m.rows = tapply(1:nrow(sprint.m),sprint.m$CityDate,find.rows,sprint.m)
sprint.m.fastest = sprint.m[sprint.m.rows,]
head(sprint.m.fastest)
```

```
##      Rank  Time Wind                 Name Country Birthdate
## 2306 2276 10.08  0.5          Bruny Surin     CAN  12.07.67
## 1249 1202 10.03 -2.1       Donovan Bailey     CAN  16.12.67
## 1599 1581 10.05  1.2      Davidson Ezinwa     NGR  22.11.71
## 2650 2532 10.09  2.0     Christie van Wyk     NAM  12.10.77
## 2297 2276 10.08  1.2    Bryan Bridgewater     USA  07.09.70
## 2119 2011 10.07  0.2 Tamunosiki Atorudibo     NGR  21.03.85
##                       City       Date                          CityDate
## 2306 &Eacute;vry-Bondoufle 11.07.1994 &Eacute;vry-Bondoufle 11.07.1994
## 1249            Abbotsford 19.07.1997            Abbotsford 19.07.1997
## 1599            Abbotsford 23.05.1992            Abbotsford 23.05.1992
## 2650               Abilene 20.05.2004               Abilene 20.05.2004
## 2297               Abilene 29.05.1993               Abilene 29.05.1993
## 2119                 Abuja 08.07.2004                 Abuja 08.07.2004
```

```
nrow(sprint.m.fastest)
```

```
## [1] 1181
```

3. Load the women's dataframe `sprint.f.csv` and repeat steps (1) and (2) on this dataset so that what remains is `sprint.f.fastest`. Display the first five rows.

```
sprint.w <- read.csv("~/Desktop/sprint.w.csv")
sprint.w$CityDate = paste(sprint.w$City, sprint.w$Date)
length(unique(sprint.w$CityDate))
```

```
## [1] 921
```

```
split.by.cd = split(sprint.w,sprint.w$CityDate)
sprint.w.rows = tapply(1:nrow(sprint.w),sprint.w$CityDate,find.rows,sprint.w)
sprint.w.fastest = sprint.w[sprint.w.rows,]
nrow(sprint.w.fastest)
```

```
## [1] 921
```

Complete the final questions only if you have time. It's not necessary for full credit.

4. We want to merge the dataframes `sprint.m.fastest` and `sprint.f.fastest` over rows that correspond to times recorded at the same track meet. First find the common track meets between the two data frames, i.e. the common entries in `CityDate`. Hint: Use `intersect()`. Call the result `common.meets`. Then compute the rows of each dataframe that correspond to these common track meets. Hint: Use `which()` and `is.element()`. Call the results `ind.m` and `ind.w`. Both should have length 385.

```
common.meets = intersect(sprint.m.fastest$CityDate,sprint.w.fastest$CityDate)
ind.m  = which(is.element(sprint.m.fastest$CityDate,common.meets))
ind.w  = which(is.element(sprint.w.fastest$CityDate,common.meets))
length(ind.m)
```

```
## [1] 385
```

```
length(ind.w)
```

```
## [1] 385
```

5. Now create a new dataframe that merges the columns of `sprint.m.fastest` with `sprint.f.fastest`, but keeping only rows that correspond to common track meets (these are indexed by `ind.m` and `ind.f`). Call the result `sprint` and arrange it so that the dataframe only has three columns: `MensTime`, `WomensTime`, and `CityDate` (the common track meet). Display the first five rows. Note here that we are implicitly assming that both `sprint.m.fastest` with `sprint.f.fastest` are ordered in the same way according to the `CityDate` variable.

```r
sprint = data.frame(sprint.m.fastest$Time[ind.m],sprint.w.fastest[ind.w,c("Time","CityDate")])
names(sprint) = c("MensTime","WomensTime","CityDate")
head(sprint)
```

```
##      MensTime WomensTime             CityDate
## 755     10.07      10.99 Ad-Dawhah 07.05.1998
## 405     10.00      10.93 Ad-Dawhah 08.05.2009
## 403     10.01      10.93 Ad-Dawhah 09.05.2008
## 361      9.87      10.92 Ad-Dawhah 11.05.2012
## 923     10.08      11.01 Ad-Dawhah 15.05.2002
## 1977     9.92      11.09      Albi 29.07.2011
```

7. Note that the previous merge could have been done with the `merge()` function. Can you get the same result using `merge()`?

```r
sprint2 = merge(sprint.m.fastest[, c("Time","CityDate")],sprint.w.fastest[,c("Time","CityDate")],by = "
names(sprint2) =c("MensTime","WomensTime","CityDate")
head(sprint2)
```

```
##               MensTime WomensTime CityDate
## 1 Ad-Dawhah 07.05.1998      10.07    10.99
## 2 Ad-Dawhah 08.05.2009      10.00    10.93
## 3 Ad-Dawhah 09.05.2008      10.01    10.93
## 4 Ad-Dawhah 11.05.2012       9.87    10.92
## 5 Ad-Dawhah 15.05.2002      10.08    11.01
## 6      Albi 29.07.2011       9.92    11.09
```