Reading and data wrangling in R

Andrea Marcela Huerfano Barbosa

August 5th, 2019

Description

In this file, you can find some tips for:

- Reading data from different formats (txt,csv,excel…)
- Cleaning data
- Creation of new variables
- Merging datasets
- Dealing with NA

All of the tasks above are related to how to clean and tidy our data, that is an inevitable phase when you work with data. Some terms for these activities are data cleaning, data wrangling, and data manipulation. ## 1. Reading data There are many ways to import datasets depending on the file characteristics as the separator, decimals, head, etc. The easy way is using the button Import Dataset in the R-Studio environment, however, you have to copy the code into your script because the lines just run in the console. To know some of the functions that appear throw the bottom you are going to find some examples.

- read.csv: comma-separated values with the period as decimal separator.
- read.csv2: semicolon-separated values with comma as decimal separator.
- read.delim: tab-delimited files with the period as decimal separator.
- read.delim2 tab-delimited files with comma as decimal separator.
- read.fwf data with a predetermined number of bytes per column.

Some functions to inspect the data are: colnames(), srt(),head(), tail()

```r
pigeon <-  read.delim("C:/Users/Andrea/Desktop/pigeon-racing.txt")
colnames(pigeon)
```

```
##  [1] "Pos"      "Breeder"  "Pigeon"   "Name"     "Color"    "Sex"
##  [7] "Ent"      "Arrival"  "Speed"    "To.Win"   "Eligible"
```

```r
str(pigeon)
```

```
## 'data.frame':    400 obs. of  11 variables:
##  $ Pos     : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Breeder : Factor w/ 90 levels "4-Birds","7-11 Syndicate",..: 83 49 47 4 40 24 40 64 9 83 ...
##  $ Pigeon  : Factor w/ 400 levels "0001-AU15-RTEX",..: 272 99 101 283 381 40 383 184 191 271 ...
##  $ Name    : Factor w/ 21 levels "","\"the Duck\"",..: 1 1 18 1 1 1 1 1 1 1 ...
##  $ Color   : Factor w/ 29 levels "BB","BBPD","BBPI",..: 9 26 1 4 6 6 5 6 1 6 ...
##  $ Sex     : Factor w/ 2 levels "C","H": 2 2 2 2 2 2 1 2 2 2 ...
##  $ Ent     : int  1 1 1 1 1 1 2 1 1 2 ...
##  $ Arrival : Factor w/ 355 levels "00:03.0","00:04.0",..: 166 183 184 185 186 188 189 190 191 192 ..
##  $ Speed   : num  172 164 163 163 163 ...
##  $ To.Win  : Factor w/ 365 levels "0:00:00","0:05:21",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Eligible: Factor w/ 1 level "Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

The summary function give you a view about distribution for cuantitative varaibles and the levels of each factor.

```r
summary(pigeon)
```

```
##       Pos                Breeder              Pigeon
##  Min.   :  1.0   Jb & D      : 13   0001-AU15-RTEX:  1
##  1st Qu.:100.8   A P C Loft  : 12   0001-IF15-POWS:  1
##  Median :200.5   Family Loft: 12   0002-AU15-RTEX:  1
##  Mean   :200.4   Redtex      : 12   0002-IF15-PJLO:  1
##  3rd Qu.:300.2   Alias-Alias: 11   0003-IF15-POWS:  1
##  Max.   :400.0   Andy Skwiat: 10   0005-AU15-NPL :  1
##                  (Other)     :330   (Other)       :394
##                Name         Color      Sex          Ent             Arrival
##                 :380   BB      :177   C:  9   Min.   : 1.000   12:20.0:  3
##  "the Duck"    :  1   BC      : 92   H:391   1st Qu.: 2.000   54:26.0:  3
##  Alice         :  1   BBWF    : 36           Median : 3.000   56:10.0:  3
##  BATTLE BORN 27:  1   RC      : 16           Mean   : 3.533   05:03.0:  2
##  Bella         :  1   DC      : 10           3rd Qu.: 5.000   07:54.0:  2
##  BLACK NIGTH 9 :  1   BCWF    :  8           Max.   :13.000   12:03.0:  2
##  (Other)       : 15   (Other): 61                             (Other):385
##      Speed             To.Win    Eligible
##  Min.   : 76.68   0:13:56:  3   Yes:400
##  1st Qu.:104.43   0:05:48:  2
##  Median :131.66   0:05:57:  2
##  Mean   :128.71   0:06:02:  2
##  3rd Qu.:151.18   0:06:41:  2
##  Max.   :172.16   0:06:48:  2
##                   (Other):387
```

excel

The functions explained above don't require installation of any library because they are in the R core, however, to read excel files it is necessary to load the library readxl.

```
library(readxl)
spanish_silver <- read_excel("C:/Users/Andrea/Desktop/spanish-silver.xls",
    sheet = "spanish-silver")
```

website

```
df <- read.table("https://s3.amazonaws.com/assets.datacamp.com/blog_assets/test.txt",
                 header = FALSE)
df
```

```
##   V1 V2 V3
## 1  1  6  a
## 2  2  7  b
## 3  3  8  c
## 4  4  9  d
## 5  5 10  e
```

Subsets

Tibble

In all of the examples above the data were loaded as data_frame. However, to display a sample of them and their visualization is easier when the data is converted into a tibble.

```r
library(tibble)
pigeon_tb <- as_data_frame(pigeon)
pigeon_tb
```

```
## # A tibble: 400 x 11
##      Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct>
##  1     1 Texas ~ 19633~ ""    BCWF  H         1 42:14.0  172. 0:00:~
##  2     2 Junior~ 0402-~ ""    SIWF  H         1 47:36.0  164. 0:05:~
##  3     3 Jerry ~ 0404-~ Perc~ BB    H         1 47:41.0  163. 0:05:~
##  4     4 Alias-~ 2013-~ ""    BBSP  H         1 47:43.0  163. 0:05:~
##  5     5 Greg G~ 5749-~ ""    BC    H         1 47:44.0  163. 0:05:~
##  6     6 Dal-Te~ 0032-~ ""    BC    H         1 47:51.0  163. 0:05:~
##  7     7 Greg G~ 5768-~ ""    BBWF  C         2 47:53.0  163. 0:05:~
##  8     8 N C Sy~ 1067-~ ""    BC    H         1 47:57.0  163. 0:05:~
##  9     9 Baldwi~ 1194-~ ""    BB    H         1 48:02.0  163. 0:05:~
## 10    10 Texas ~ 19632~ ""    BC    H         2 48:03.0  163. 0:05:~
## # ... with 390 more rows, and 1 more variable: Eligible <fct>
```

This sort of view is obtained directly into the original dataframe with the function head.

```r
head(pigeon, n=4)
```

```
##   Pos            Breeder           Pigeon         Name Color Sex Ent Arrival
## 1   1      Texas Outlaws 19633-AU15-FOYS               BCWF   H   1 42:14.0
## 2   2     Junior Juanich   0402-AU15-JRL               SIWF   H   1 47:36.0
## 3   3 Jerry Allensworth  0404-AU15-VITA Perch Potato    BB   H   1 47:41.0
## 4   4        Alias-Alias  2013-AU15-ALIA               BBSP   H   1 47:43.0
##     Speed  To.Win Eligible
## 1 172.155 0:00:00      Yes
## 2 163.569 0:05:21      Yes
## 3 163.442 0:05:27      Yes
## 4 163.392 0:05:28      Yes
```

In this script most of the data will be used in tibbles.

Sampling

After loading the dataset is useful sampling to know their data and identify steps to clean them.

```r
library(dplyr)
pigeon_tb%>%sample_n(4)
```

```
## # A tibble: 4 x 11
##     Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##   <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct>
## 1    56 Shang ~ 0929-~ ""    BBSP  H         1 50:24.0  159. 0:08:~
## 2    99 Sierra~ 0535-~ ""    BB    H         3 56:15.0  151. 0:14:~
## 3   110 Milner~ 2483-~ ""    BB    H         2 56:55.0  151. 0:14:~
## 4   223 Mc Lau~ 2117-~ ""    BB    H         3 24:20.0  122. 0:42:~
## # ... with 1 more variable: Eligible <fct>
```

Extracting a percentage in the data set

```r
pigeon_tb%>%sample_frac(0.01, replace=FALSE)
```

```
## # A tibble: 4 x 11
##     Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##   <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct>
## 1   182 Redtex  0007-~ ""    RC    H         9 07:39.0 138.  0:25:~
## 2   266 Woodse~ 1535-~ ""    BCWF  H         7 40:17.0 110.  0:58:~
## 3    42 Dave H~ 0009-~ ""    BC    H         1 49:30.0 161.  0:07:~
## 4   336 4-Birds 0760-~ ""    BB    H         3 57:26.0  99.2 1:15:~
## # ... with 1 more variable: Eligible <fct>
```

Selecting columns

```
pigeon_tb%>%select(Pigeon, Color, Sex)
```

```
## # A tibble: 400 x 3
##    Pigeon          Color Sex  
##    <fct>           <fct> <fct>
##  1 19633-AU15-FOYS BCWF  H    
##  2 0402-AU15-JRL   SIWF  H    
##  3 0404-AU15-VITA  BB    H    
##  4 2013-AU15-ALIA  BBSP  H    
##  5 5749-AU15-SLI   BC    H    
##  6 0032-AU15-DRPC  BC    H    
##  7 5768-AU15-SLI   BBWF  C    
##  8 1067-AU15-TXHC  BC    H    
##  9 1194-AU15-TENT  BB    H    
## 10 19632-AU15-FOYS BC    H    
## # ... with 390 more rows
```

Filters

- And &
- Or |

```
pigeon_tb%>%filter(Color=='BB'  | Sex=='H')
```

```
## # A tibble: 396 x 11
##      Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct> 
##  1     1 Texas ~ 19633~ ""    BCWF  H         1 42:14.0  172. 0:00:~
##  2     2 Junior~ 0402-~ ""    SIWF  H         1 47:36.0  164. 0:05:~
##  3     3 Jerry ~ 0404-~ Perc~ BB    H         1 47:41.0  163. 0:05:~
##  4     4 Alias-~ 2013-~ ""    BBSP  H         1 47:43.0  163. 0:05:~
##  5     5 Greg G~ 5749-~ ""    BC    H         1 47:44.0  163. 0:05:~
##  6     6 Dal-Te~ 0032-~ ""    BC    H         1 47:51.0  163. 0:05:~
##  7     8 N C Sy~ 1067-~ ""    BC    H         1 47:57.0  163. 0:05:~
##  8     9 Baldwi~ 1194-~ ""    BB    H         1 48:02.0  163. 0:05:~
##  9    10 Texas ~ 19632~ ""    BC    H         2 48:03.0  163. 0:05:~
## 10    10 Redtex  0024-~ ""    RED   H         1 48:03.0  163. 0:05:~
## # ... with 386 more rows, and 1 more variable: Eligible <fct>
```

```
pigeon_tb%>%filter(Color=='BB'  & Sex=='H')
```

```
## # A tibble: 172 x 11
##      Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct> 
##  1     3 Jerry ~ 0404-~ Perc~ BB    H         1 47:41.0  163. 0:05:~
##  2     9 Baldwi~ 1194-~ ""    BB    H         1 48:02.0  163. 0:05:~
##  3    14 Goshen~ 5834-~ ""    BB    H         1 48:12.0  163. 0:05:~
## 4     16 Flyhom~ 1531-~ ""     BB    H         1 48:15.0  163. 0:06:~
## 5     24 Jb & D  1214-~ ""     BB    H         1 48:36.0  162. 0:06:~
## 6     30 Churn ~ 9216-~ ""     BB    H         1 48:48.0  162. 0:06:~
## 7     32 Alias-~ 2049-~ ""     BB    H         3 48:56.0  162. 0:06:~
## 8     35 Clear ~ 0263-~ ""     BB    H         1 49:06.0  161. 0:06:~
## 9     38 Clear ~ 0235-~ ""     BB    H         2 49:17.0  161. 0:07:~
## 10    40 Skip's~ 5302-~ ""     BB    H         2 49:28.0  161. 0:07:~
## # ... with 162 more rows, and 1 more variable: Eligible <fct>
```

Order by

The "-" makes the order from the greearest to the shortest.

```
pigeon_tb%>%arrange(-Speed)
```

```
## # A tibble: 400 x 11
##      Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct>
## 1      1 Texas ~ 19633~ ""    BCWF  H         1 42:14.0  172. 0:00:~
## 2      2 Junior~ 0402-~ ""    SIWF  H         1 47:36.0  164. 0:05:~
## 3      3 Jerry ~ 0404-~ Perc~ BB    H         1 47:41.0  163. 0:05:~
## 4      4 Alias-~ 2013-~ ""    BBSP  H         1 47:43.0  163. 0:05:~
## 5      5 Greg G~ 5749-~ ""    BC    H         1 47:44.0  163. 0:05:~
## 6      6 Dal-Te~ 0032-~ ""    BC    H         1 47:51.0  163. 0:05:~
## 7      7 Greg G~ 5768-~ ""    BBWF  C         2 47:53.0  163. 0:05:~
## 8      8 N C Sy~ 1067-~ ""    BC    H         1 47:57.0  163. 0:05:~
## 9      9 Baldwi~ 1194-~ ""    BB    H         1 48:02.0  163. 0:05:~
## 10    10 Texas ~ 19632~ ""    BC    H         2 48:03.0  163. 0:05:~
## # ... with 390 more rows, and 1 more variable: Eligible <fct>
```

2.Cleaning data

Creation of new variables

New variable

```
pigeon_tb%>%mutate(NewSpeed=Speed/2)
```

```
## # A tibble: 400 x 12
##      Pos Breeder Pigeon Name  Color Sex     Ent Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <int> <fct>   <dbl> <fct>
## 1      1 Texas ~ 19633~ ""    BCWF  H         1 42:14.0  172. 0:00:~
## 2      2 Junior~ 0402-~ ""    SIWF  H         1 47:36.0  164. 0:05:~
## 3      3 Jerry ~ 0404-~ Perc~ BB    H         1 47:41.0  163. 0:05:~
## 4      4 Alias-~ 2013-~ ""    BBSP  H         1 47:43.0  163. 0:05:~
## 5      5 Greg G~ 5749-~ ""    BC    H         1 47:44.0  163. 0:05:~
## 6      6 Dal-Te~ 0032-~ ""    BC    H         1 47:51.0  163. 0:05:~
## 7      7 Greg G~ 5768-~ ""    BBWF  C         2 47:53.0  163. 0:05:~
## 8      8 N C Sy~ 1067-~ ""    BC    H         1 47:57.0  163. 0:05:~
## 9      9 Baldwi~ 1194-~ ""    BB    H         1 48:02.0  163. 0:05:~
## 10    10 Texas ~ 19632~ ""    BC    H         2 48:03.0  163. 0:05:~
## # ... with 390 more rows, and 2 more variables: Eligible <fct>,
## #   NewSpeed <dbl>
```

Split

Split a string by an specific separator.

```r
library(dplyr)
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```r
pigeon_tb%>%separate(Pigeon, sep='-', c('Num', 'id', 'det'))
```

```
## # A tibble: 400 x 13
##      Pos Breeder Num   id    det   Name  Color Sex     Ent Arrival Speed
##    <int> <fct>   <chr> <chr> <chr> <fct> <fct> <fct> <int> <fct>   <dbl>
##  1     1 Texas ~ 19633 AU15  FOYS  ""    BCWF  H         1 42:14.0  172.
##  2     2 Junior~ 0402  AU15  JRL   ""    SIWF  H         1 47:36.0  164.
##  3     3 Jerry ~ 0404  AU15  VITA  Perc~ BB    H         1 47:41.0  163.
##  4     4 Alias-~ 2013  AU15  ALIA  ""    BBSP  H         1 47:43.0  163.
##  5     5 Greg G~ 5749  AU15  SLI   ""    BC    H         1 47:44.0  163.
##  6     6 Dal-Te~ 0032  AU15  DRPC  ""    BC    H         1 47:51.0  163.
##  7     7 Greg G~ 5768  AU15  SLI   ""    BBWF  C         2 47:53.0  163.
##  8     8 N C Sy~ 1067  AU15  TXHC  ""    BC    H         1 47:57.0  163.
##  9     9 Baldwi~ 1194  AU15  TENT  ""    BB    H         1 48:02.0  163.
## 10    10 Texas ~ 19632 AU15  FOYS  ""    BC    H         2 48:03.0  163.
## # ... with 390 more rows, and 2 more variables: To.Win <fct>,
## #   Eligible <fct>
```

Concatenate

```r
pigeon_tb%>%unite_('new', c('Pos','Sex'), sep = '-')
```

```
## # A tibble: 400 x 10
##    new   Breeder   Pigeon  Name   Color   Ent Arrival Speed To.Win Eligible
##    <chr> <fct>     <fct>   <fct>  <fct> <int> <fct>   <dbl> <fct>  <fct>
##  1 1-H   Texas Ou~ 19633-~ ""     BCWF      1 42:14.0  172. 0:00:~ Yes
##  2 2-H   Junior J~ 0402-A~ ""     SIWF      1 47:36.0  164. 0:05:~ Yes
##  3 3-H   Jerry Al~ 0404-A~ Perch~ BB        1 47:41.0  163. 0:05:~ Yes
##  4 4-H   Alias-Al~ 2013-A~ ""     BBSP      1 47:43.0  163. 0:05:~ Yes
##  5 5-H   Greg Gla~ 5749-A~ ""     BC        1 47:44.0  163. 0:05:~ Yes
##  6 6-H   Dal-Tex ~ 0032-A~ ""     BC        1 47:51.0  163. 0:05:~ Yes
##  7 7-C   Greg Gla~ 5768-A~ ""     BBWF      2 47:53.0  163. 0:05:~ Yes
##  8 8-H   N C Synd~ 1067-A~ ""     BC        1 47:57.0  163. 0:05:~ Yes
##  9 9-H   Baldwin ~ 1194-A~ ""     BB        1 48:02.0  163. 0:05:~ Yes
## 10 10-H  Texas Ou~ 19632-~ ""     BC        2 48:03.0  163. 0:05:~ Yes
## # ... with 390 more rows
```

New varaible base on levels of another one

```r
levels(pigeon_tb$Color)
```

```
##  [1] "BB"   "BBPD" "BBPI" "BBSP" "BBWF" "BC"   "BCH"  "BCSP" "BCWF" "BKWF"
## [11] "BLCK" "BLK"  "DC"   "DCWF" "GRIZ" "GRZL" "OPAL" "OPWF" "PENC" "RC"
## [21] "RCSP" "RCWF" "RED"  "SIL"  "SILV" "SIWF" "WGRZ" "WHGR" "WHT"
```

```r
B_I<-c("BB","BBPD","BBPI","BBSP","BBWF","BC","BCH","BCSP","BCWF","BKWF","BLCK","BLK")
D_I<-c("DC","DCWF")
G_I<-c("GRIZ","GRZL")

for (i  in 1:length(pigeon_tb$Color)){
  if  (pigeon_tb$Color[i] %in% B_I){pigeon_tb$Ini[i]='B_I'}else{
    if (pigeon_tb$Color[i] %in% D_I){pigeon_tb$Ini[i]='D_I'}else{
      if(pigeon_tb$Color[i] %in% G_I){pigeon_tb$Ini[i]='G_I'}else{pigeon_tb$Ini[i]='Another'}
      }
    }
}
```

```
## Warning: Unknown or uninitialised column: 'Ini'.
```

```
as_data_frame(data.frame(pigeon_tb$Color,pigeon_tb$Ini))
```

```
## # A tibble: 400 x 2
##    pigeon_tb.Color pigeon_tb.Ini
##    <fct>           <fct>
##  1 BCWF            B_I
##  2 SIWF            Another
##  3 BB              B_I
##  4 BBSP            B_I
##  5 BC              B_I
##  6 BC              B_I
##  7 BBWF            B_I
##  8 BC              B_I
##  9 BB              B_I
## 10 BC              B_I
## # ... with 390 more rows
```

Variable type conversion

Supose that Ent is a factor variable not a numeric one.

```
pigeon_tb$Ent<- as.factor(pigeon_tb$Ent)
pigeon_tb
```

```
## # A tibble: 400 x 12
##      Pos Breeder Pigeon Name  Color Sex   Ent   Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <fct> <fct>   <dbl> <fct>
##  1     1 Texas ~ 19633~ ""    BCWF  H     1     42:14.0  172. 0:00:~
##  2     2 Junior~ 0402-~ ""    SIWF  H     1     47:36.0  164. 0:05:~
##  3     3 Jerry ~ 0404-~ Perc~ BB    H     1     47:41.0  163. 0:05:~
##  4     4 Alias-~ 2013-~ ""    BBSP  H     1     47:43.0  163. 0:05:~
##  5     5 Greg G~ 5749-~ ""    BC    H     1     47:44.0  163. 0:05:~
##  6     6 Dal-Te~ 0032-~ ""    BC    H     1     47:51.0  163. 0:05:~
##  7     7 Greg G~ 5768-~ ""    BBWF  C     2     47:53.0  163. 0:05:~
##  8     8 N C Sy~ 1067-~ ""    BC    H     1     47:57.0  163. 0:05:~
##  9     9 Baldwi~ 1194-~ ""    BB    H     1     48:02.0  163. 0:05:~
## 10    10 Texas ~ 19632~ ""    BC    H     2     48:03.0  163. 0:05:~
## # ... with 390 more rows, and 2 more variables: Eligible <fct>, Ini <chr>
```

If the variable is as string to convert them type into numeric the function is as.numeric()

Commonly, you have to merge many files to obtain your final dataset. In R at the same that Python you need to have the same colname in the key variable.

Joins

R has the SQL functions to join files, the key to join the data sets must have the same name in the files.

```
library(readxl)
athlete_country <- read_excel("C:/Users/Andrea/Desktop/python-ml-course-master/datasets/athletes/athlet
    sheet = "Athelete_Country_Map")

athlete_sport <- read_excel("C:/Users/Andrea/Desktop/python-ml-course-master/datasets/athletes/athlete.
    sheet = "Athelete")

athlete_country
```

```
## # A tibble: 6,970 x 2
##    Athlete           Country
##    <chr>             <chr>
##  1 Michael Phelps    United States
##  2 Natalie Coughlin United States
##  3 Aleksey Nemov     Russia
##  4 Alicia Coutts     Australia
##  5 Missy Franklin    United States
##  6 Ryan Lochte       United States
##  7 Allison Schmitt   United States
##  8 Ian Thorpe        Australia
##  9 Dara Torres       United States
## 10 Cindy Klassen     Canada
## # ... with 6,960 more rows
```

```
athlete_sport
```

```
## # A tibble: 6,975 x 2
##    Athlete           Sport
##    <chr>             <chr>
##  1 Michael Phelps    Swimming
##  2 Natalie Coughlin Swimming
##  3 Aleksey Nemov     Gymnastics
##  4 Alicia Coutts     Swimming
##  5 Missy Franklin    Swimming
##  6 Ryan Lochte       Swimming
##  7 Allison Schmitt   Swimming
##  8 Ian Thorpe        Swimming
##  9 Dara Torres       Swimming
## 10 Cindy Klassen     Speed Skating
## # ... with 6,965 more rows
```

For this example the key is the column called ‘Athlete’

```
inner_join(athlete_country, athlete_sport, by='Athlete')
```

```
## # A tibble: 6,994 x 3
##    Athlete           Country       Sport
##    <chr>             <chr>         <chr>
##  1 Michael Phelps    United States Swimming
##  2 Natalie Coughlin United States Swimming
##  3 Aleksey Nemov     Russia        Gymnastics
##  4 Alicia Coutts     Australia     Swimming
##  5 Missy Franklin    United States Swimming
##  6 Ryan Lochte       United States Swimming
##  7 Allison Schmitt   United States Swimming
##  8 Ian Thorpe        Australia     Swimming
##  9 Dara Torres      United States Swimming
## 10 Cindy Klassen    Canada        Speed Skating
## # ... with 6,984 more rows
```

The structure to reproduce left and right join is the same that the example above.

Matching strings

There are two ways to match strings, the first one is creating a list of all levels and defining the category when each one belongs, the second way is defining a distance between two strings base on how different they are.

Uppercase

The 'M' doesn't match with 'm', first of all is necessary to homogenize the strings, for example, all of them in uppercase.

```
library(R.utils)
pigeon_tb%>%mutate(Breeder=toupper(Breeder))
```

```
## # A tibble: 400 x 12
##      Pos Breeder Pigeon Name  Color Sex   Ent   Arrival Speed To.Win
##    <int> <chr>   <fct>  <fct> <fct> <fct> <fct> <fct>   <dbl> <fct>
##  1     1 TEXAS ~ 19633~ ""    BCWF  H     1     42:14.0  172. 0:00:~
##  2     2 JUNIOR~ 0402-~ ""    SIWF  H     1     47:36.0  164. 0:05:~
##  3     3 JERRY ~ 0404-~ Perc~ BB    H     1     47:41.0  163. 0:05:~
##  4     4 ALIAS-~ 2013-~ ""    BBSP  H     1     47:43.0  163. 0:05:~
##  5     5 GREG G~ 5749-~ ""    BC    H     1     47:44.0  163. 0:05:~
##  6     6 DAL-TE~ 0032-~ ""    BC    H     1     47:51.0  163. 0:05:~
##  7     7 GREG G~ 5768-~ ""    BBWF  C     2     47:53.0  163. 0:05:~
##  8     8 N C SY~ 1067-~ ""    BC    H     1     47:57.0  163. 0:05:~
##  9     9 BALDWI~ 1194-~ ""    BB    H     1     48:02.0  163. 0:05:~
## 10    10 TEXAS ~ 19632~ ""    BC    H     2     48:03.0  163. 0:05:~
## # ... with 390 more rows, and 2 more variables: Eligible <fct>, Ini <chr>
```

Identifing if a substring is inside another one

```
gender <- c("MA", "male ", "Female", "fem.", 'ma', 'Fe')
grepl("ma", gender)
```

```
## [1] FALSE  TRUE  TRUE FALSE  TRUE FALSE
```

Ignoring upper and lower case

```
grepl("m", gender, ignore.case = TRUE)
```

```
## [1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
```

Just starting with m

```
grepl("^m", gender, ignore.case = TRUE)
```

```
## [1]  TRUE  TRUE FALSE FALSE  TRUE FALSE
```

String distances

String distances is an algorithm that identify how much different two strings are. This example is part of the Van der Loo, M. and De Jonge, E. (2013) in page 25 if you want go deep in details.

```
codes <- c("male", "female")
D <- adist(gender, codes)
```

```
colnames(D) <- codes
rownames(D) <- gender
D <- transform(D, min = pmin(male, female)) #however this function just return the value
D
```

```
##        male female min
## MA        4      6   4
## male      1      3   1
## Female    2      1   1
## fem.      4      3   3
## ma        2      4   2
## Fe        3      5   3
```

Identifying the column which contain the minimun distance

```
i <- apply(D, 1, which.min)
i
```

```
##     MA   male  Female    fem.      ma      Fe
##      1      1      2      2      1      1
```

```
data.frame(rawtext = gender, coded = codes[i])
```

```
##   rawtext  coded
## 1      MA   male
## 2   male    male
## 3  Female female
## 4    fem. female
## 5      ma   male
## 6      Fe   male
```

5. Dealing with NA

Counting the na values

```
sapply(pigeon_tb, function(x) sum(is.na(x)))
```

```
##      Pos  Breeder   Pigeon     Name    Color      Sex      Ent  Arrival
##        0        0        0        0        0        0        0        0
##    Speed   To.Win Eligible      Ini
##        0        0        0        0
```

This is weird especially when I new that in name there are too many rows in blank, then one of the levels of the variable must be “”

```
levels(pigeon_tb$Name)
```

```
##  [1] ""               "\"the Duck\""   "Alice"          "BATTLE BORN 27"
##  [5] "Bella"          "BLACK NIGTH 9"  "Canned Heat"    "Charlie"
##  [9] "Christie"       "Color Me Hot"   "Edward"         "Elle"
## [13] "Gage"           "Gypsy"          "Jack Frost"     "Kingston"
## [17] "Lil Dat"        "Perch Potato"   "Pop's Pick"     "Rogue Brew"
## [21] "SEMPER FI 11"
```

The level “” is defining as NA

```
levels(pigeon_tb$Name)[levels(pigeon_tb$Name)==""]<-NA
levels(pigeon_tb$Name)
```

```
##  [1] "\"the Duck\""   "Alice"          "BATTLE BORN 27" "Bella"
##  [5] "BLACK NIGTH 9"  "Canned Heat"    "Charlie"        "Christie"
##  [9] "Color Me Hot"   "Edward"         "Elle"           "Gage"
## [13] "Gypsy"          "Jack Frost"     "Kingston"       "Lil Dat"
## [17] "Perch Potato"   "Pop's Pick"     "Rogue Brew"     "SEMPER FI 11"
```

```
pigeon_tb
```

```
## # A tibble: 400 x 12
##      Pos Breeder Pigeon Name  Color Sex   Ent   Arrival Speed To.Win
##    <int> <fct>   <fct>  <fct> <fct> <fct> <fct> <fct>   <dbl> <fct>
##  1     1 Texas ~ 19633~ <NA>  BCWF  H     1     42:14.0  172. 0:00:~
##  2     2 Junior~ 0402-~ <NA>  SIWF  H     1     47:36.0  164. 0:05:~
##  3     3 Jerry ~ 0404-~ Perc~ BB    H     1     47:41.0  163. 0:05:~
##  4     4 Alias-~ 2013-~ <NA>  BBSP  H     1     47:43.0  163. 0:05:~
##  5     5 Greg G~ 5749-~ <NA>  BC    H     1     47:44.0  163. 0:05:~
##  6     6 Dal-Te~ 0032-~ <NA>  BC    H     1     47:51.0  163. 0:05:~
##  7     7 Greg G~ 5768-~ <NA>  BBWF  C     2     47:53.0  163. 0:05:~
##  8     8 N C Sy~ 1067-~ <NA>  BC    H     1     47:57.0  163. 0:05:~
##  9     9 Baldwi~ 1194-~ <NA>  BB    H     1     48:02.0  163. 0:05:~
## 10    10 Texas ~ 19632~ <NA>  BC    H     2     48:03.0  163. 0:05:~
## # ... with 390 more rows, and 2 more variables: Eligible <fct>, Ini <chr>
```

Imputation is a very imporant topic and needs go in severals details for that reason it is not covered in this paper.

References

Van der Loo, M. and De Jonge, E. (2013) An introduction to data cleaning with R. https://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf

Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2019). dplyr: A Grammar of Data Manipulation. R package version 0.8.3. https://CRAN.R-project.org/package=dplyr

Hadley Wickham and Lionel Henry (2019). tidyr: Easily Tidy Data with 'spread()' and 'gather()' Functions. R package version 0.8.3. https://CRAN.R-project.org/package=tidyr

Kirill Müller and Hadley Wickham (2019). tibble: Simple Data Frames. R package version 2.1.3. https://CRAN.R-project.org/package=tibble

Hadley Wickham and Jennifer Bryan (2019). readxl: Read Excel Files. R package version 1.3.1. https://CRAN.R-project.org/package=readxl

Hadley Wickham (2017). tidyverse: Easily Install and Load the 'Tidyverse'. R package version 1.2.1. https://CRAN.R-project.org/package=tidyverse

Usefull resources

- Presentation data cleaning Jonge. https://www.r-project.ro/conference2017/presentations/uRos2017_data-cleaning-workshop.pdf