Solution Data Cleaning

Giovanna de Vincenzo

22 luglio, 2019

Contents

Load required packages

```
library(readr)
library(dplyr)
library(stringr)
library(reshape)
```

Import dataset

```
markets <- read.csv("Farmers_Markets.csv")
class(markets)
```

```
## [1] "data.frame"
```

Inspect your data

First six rows of dataset

```
head(markets)
```

```
##        Ã¯..X        Y OBJECTID                         NEIGHBORHOOD
## 1 -75.12156 39.97843        1 Bridesburg Kensington Port Richmond
## 2 -75.13365 39.98297        2 Bridesburg Kensington Port Richmond
## 3 -75.10947 39.98255        3 Bridesburg Kensington Port Richmond
## 4 -75.17067 39.95014        4                          Center City
## 5 -75.18114 39.94902        5                         Center City
## 6 -75.17399 39.96732        6                          Center City
##                                  NAME                       ADDRESS
## 1              Greensgrow Farm Stand          2501 E. Cumberland Ave
## 2 Norris Square Neighborhood Project               2141 N. Howard St
## 3                        Powers Park            Almond and E. Ann St
## 4                        Rittenhouse             18th and Walnut St.
## 5              Schuylkill River Park              25th and Spruce St
## 6                          Fairmount N. 22nd St. and Fairmount Ave.
```

```
##                            ADDRESS_NOTES        DAY
## 1                                           Mon-Sun
## 2 Between Norris St and Susquehanna Ave        Wed
## 3                                            Thurs
## 4                                        Tues & Sat
## 5                                            Wed
## 6                                            Thurs
##                                  TIME
## 1 Mon-Fri 10am-5pm, Sat-Sun 10am-4pm
## 2                            11am-1pm
## 3                              3-7pm
## 4          Tues 10am-2pm, Sat 9am-3pm
## 5                              3-7pm
## 6                              3-7pm
##                                   MONTHS ACCEPT_SNAP_ACCESS ACCEPT_FMNP
## 1                    Late May - November                  Y           Y
## 2                         May - November            Applied     Applied
## 3                        May - November                              Y
## 4 Tues- June - November; Sat- Year round                             Y
## 5                 May 17th - late October                            Y
## 6                     June 1st - November                  Y           Y
##   ACCEPT_PHILLY_FOOD_BUCKS_ MAJOR_BUS_SUBWAY_ROUTES
## 1                        NA          25, 39, 43, 89
## 2                        NA               3, 39, 89
## 3                        NA             15B, 54, 60
## 4                        NA                   9, 17
## 5                        NA                7, 12, 40
## 6                        NA   7, 32, 33, 48, PHLASH
```

Variable names

```
names(markets)
```

```
##  [1] "Ã¯..X"                     "Y"
##  [3] "OBJECTID"                  "NEIGHBORHOOD"
##  [5] "NAME"                      "ADDRESS"
##  [7] "ADDRESS_NOTES"             "DAY"
##  [9] "TIME"                      "MONTHS"
## [11] "ACCEPT_SNAP_ACCESS"        "ACCEPT_FMNP"
## [13] "ACCEPT_PHILLY_FOOD_BUCKS_" "MAJOR_BUS_SUBWAY_ROUTES"
```

Dimensions of data frame

```
dim(markets)
```

```
## [1] 54 14
```

How many NAs do we have per variable

```
summary(markets)
```

```
##       Ã¯..X              Y            OBJECTID
##  Min.   :-75.25   Min.   :39.92   Min.   : 1.00
##  1st Qu.:-75.21   1st Qu.:39.95   1st Qu.:14.25
##  Median :-75.17   Median :39.98   Median :27.50
##  Mean   :-75.17   Mean   :39.98   Mean   :27.50
##  3rd Qu.:-75.15   3rd Qu.:40.01   3rd Qu.:40.75
##  Max.   :-75.04   Max.   :40.07   Max.   :54.00
```

```
##
##                     NEIGHBORHOOD                  NAME
##  North                    : 9    11th & Dauphin    : 1
##  Center City              : 8    22nd & Tasker     : 1
##  West                     : 6    26th and Allegheny: 1
##  West                     : 6    33rd & Diamond    : 1
##  South                    : 5    42nd and Girard   : 1
##  Germantown Chestnut Hill: 4    4th and Lehigh    : 1
##  (Other)                  :16    (Other)           :48
##                      ADDRESS                                    ADDRESS_NOTES
##  12th and Arch St       : 2                                          :38
##  10th and Chestnut St.  : 1   At Saul High School                   : 1
##  18th and Walnut St.    : 1   At the entrance to Bartram's Garden    : 1
##  2141 N. Howard St      : 1   At the Overbrook Presbyterian Church: 1
##  2144 Cecil B. Moore Ave: 1   At the West Philly YMCA                : 1
##  215 E. Penn St         : 1   Between 49th and 50th Streets          : 1
##  (Other)                :47   (Other)                                :11
##              DAY           TIME                MONTHS
##  Sat           :12   2-6pm   :10   May - November : 4
##  Wed           :11   10am-2pm: 8                 : 3
##  Thurs         : 9   3-7pm   : 6   May - November : 3
##  Tues          : 5   1-5pm   : 2   May- November  : 2
##  Mon - Sat, Sun: 2   11am-2pm: 2   Open year round: 2
##  Sun           : 2   11am-3pm: 2   Year round     : 2
##  (Other)       :13   (Other) :24   (Other)        :38
##     ACCEPT_SNAP_ACCESS  ACCEPT_FMNP ACCEPT_PHILLY_FOOD_BUCKS_
##              :11              : 7   Mode:logical
##              : 1       Applied: 1   NA's:54
##  Applied     : 1       Y      :46
##  McCanns Farm: 1
##  Y           :39
##  Y           : 1
##
##   MAJOR_BUS_SUBWAY_ROUTES
##               : 4
##  23           : 2
##  7, 12, 40    : 2
##  12, 40, 57   : 1
##  13 trolley, G: 1
##  13, 30, 34   : 1
##  (Other)      :43
```

Data-viewer

```
View(markets)
```

Display internal structure

```
str(markets)
```

```
## 'data.frame':    54 obs. of  14 variables:
##  $ Ã¯..X                    : num  -75.1 -75.1 -75.1 -75.2 -75.2 ...
##  $ Y                        : num  40 40 40 40 39.9 ...
##  $ OBJECTID                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ NEIGHBORHOOD             : Factor w/ 16 levels "Bridesburg Kensington Port Richmond",..: 1 1 1 3 4
##  $ NAME                     : Factor w/ 54 levels "11th & Dauphin",..: 23 39 44 47 49 17 32 18 27 46
```

```
##  $ ADDRESS                  : Factor w/ 53 levels "10th and Chestnut St.",..: 9 4 37 3 11 47 1 8 14 1
##  $ ADDRESS_NOTES            : Factor w/ 17 levels "","At Saul High School",..: 1 7 1 1 1 1 1 1 1 .
##  $ DAY                      : Factor w/ 19 levels "1st & 3rd Fri",..: 8 19 13 16 19 13 13 11 12 9 ..
##  $ TIME                     : Factor w/ 27 levels "1-5pm","10am-2pm",..: 22 5 14 27 14 14 7 20 2 23
##  $ MONTHS                   : Factor w/ 44 levels "","July - Sept",..: 21 26 27 43 32 16 27 40 40 44
##  $ ACCEPT_SNAP_ACCESS       : Factor w/ 6 levels "","  ","Applied",..: 5 3 1 1 2 5 4 5 5 5 ...
##  $ ACCEPT_FMNP              : Factor w/ 3 levels "","Applied","Y": 3 2 3 3 3 3 3 3 3 3 ...
##  $ ACCEPT_PHILLY_FOOD_BUCKS_: logi  NA NA NA NA NA NA ...
##  $ MAJOR_BUS_SUBWAY_ROUTES  : Factor w/ 49 levels "","12, 40, 57",..: 15 18 6 41 37 38 48 37 2 23 ..
```

Dealing with missing values

```
#replace blanks/spaces with NAs
markets[markets==""| markets==" "] = NA

#for factor variables:
markets <- markets %>%
  mutate_if(is.factor, funs(factor(replace(., .=="" | .==" ", NA))))

#Question: What do you do when there are text NAs where actual NAs should be?

markets[markets=="NA"]<-NA
```

We can solve all of the problems above by changing the arguments when importing data:

```
markets <- read.csv("Farmers_Markets.csv", na.strings=c("NA","NaN", " ", ""))
str(markets)
```

```
## 'data.frame':    54 obs. of  14 variables:
##  $ Ã¯..X                   : num  -75.1 -75.1 -75.1 -75.2 -75.2 ...
##  $ Y                       : num  40 40 40 40 39.9 ...
##  $ OBJECTID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ NEIGHBORHOOD            : Factor w/ 16 levels "Bridesburg Kensington Port Richmond",..: 1 1 1 3 4
##  $ NAME                    : Factor w/ 54 levels "11th & Dauphin",..: 23 39 44 47 49 17 32 18 27 46
##  $ ADDRESS                 : Factor w/ 53 levels "10th and Chestnut St.",..: 9 4 37 3 11 47 1 8 14 1
##  $ ADDRESS_NOTES           : Factor w/ 16 levels "At Saul High School",..: NA 6 NA NA NA NA NA NA NA
##  $ DAY                     : Factor w/ 19 levels "1st & 3rd Fri",..: 8 19 13 16 19 13 13 11 12 9 ..
##  $ TIME                    : Factor w/ 27 levels "1-5pm","10am-2pm",..: 22 5 14 27 14 14 7 20 2 23
##  $ MONTHS                  : Factor w/ 43 levels "July - Sept",..: 20 25 26 42 31 15 26 39 39 43 ..
##  $ ACCEPT_SNAP_ACCESS      : Factor w/ 4 levels "Applied","McCanns Farm",..: 3 1 NA NA NA 3 2 3 3 3
##  $ ACCEPT_FMNP             : Factor w/ 2 levels "Applied","Y": 2 1 2 2 2 2 2 2 2 2 ...
##  $ ACCEPT_PHILLY_FOOD_BUCKS_: logi  NA NA NA NA NA NA ...
##  $ MAJOR_BUS_SUBWAY_ROUTES  : Factor w/ 48 levels "12, 40, 57","13 trolley, G",..: 14 17 5 40 36 37 4
```

Variable types

Find out what class each variable in the dataset is

```
str(markets)
```

```
## 'data.frame':    54 obs. of  14 variables:
##  $ Ã¯..X                   : num  -75.1 -75.1 -75.1 -75.2 -75.2 ...
##  $ Y                       : num  40 40 40 40 39.9 ...
##  $ OBJECTID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ NEIGHBORHOOD            : Factor w/ 16 levels "Bridesburg Kensington Port Richmond",..: 1 1 1 3 4
##  $ NAME                    : Factor w/ 54 levels "11th & Dauphin",..: 23 39 44 47 49 17 32 18 27 46
```

```
##  $ ADDRESS                  : Factor w/ 53 levels "10th and Chestnut St.",..: 9 4 37 3 11 47 1 8 14 1
##  $ ADDRESS_NOTES            : Factor w/ 16 levels "At Saul High School",..: NA 6 NA NA NA NA NA NA NA
##  $ DAY                      : Factor w/ 19 levels "1st & 3rd Fri",..: 8 19 13 16 19 13 13 11 12 9 ..
##  $ TIME                     : Factor w/ 27 levels "1-5pm","10am-2pm",..: 22 5 14 27 14 14 7 20 2 23
##  $ MONTHS                   : Factor w/ 43 levels "July - Sept",..: 20 25 26 42 31 15 26 39 39 43 ..
##  $ ACCEPT_SNAP_ACCESS       : Factor w/ 4 levels "Applied","McCanns Farm",..: 3 1 NA NA NA 3 2 3 3 3
##  $ ACCEPT_FMNP              : Factor w/ 2 levels "Applied","Y": 2 1 2 2 2 2 2 2 2 2 ...
##  $ ACCEPT_PHILLY_FOOD_BUCKS_: logi  NA NA NA NA NA NA ...
##  $ MAJOR_BUS_SUBWAY_ROUTES  : Factor w/ 48 levels "12, 40, 57","13 trolley, G",..: 14 17 5 40 36 37 4
```

```
#could also use
#lapply(markets, class)
```

why don't we have any character variables? default setting for read.csv is to import strings as factors to change this, and reimport the dataset with strings as characters we could run:

```
markets <- read.csv("Farmers_Markets.csv", stringsAsFactors = F, na.strings=c("NA","NaN", " ", ""))
str(markets)
```

```
## 'data.frame':    54 obs. of  14 variables:
##  $ Ã¯..X                    : num  -75.1 -75.1 -75.1 -75.2 -75.2 ...
##  $ Y                        : num  40 40 40 40 39.9 ...
##  $ OBJECTID                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ NEIGHBORHOOD             : chr  "Bridesburg Kensington Port Richmond" "Bridesburg Kensington Port
##  $ NAME                     : chr  "Greensgrow Farm Stand" "Norris Square Neighborhood Project" "Powe
##  $ ADDRESS                  : chr  "2501 E. Cumberland Ave" "2141 N. Howard St" "Almond and E. Ann St
##  $ ADDRESS_NOTES            : chr  NA "Between Norris St and Susquehanna Ave" NA NA ...
##  $ DAY                      : chr  "Mon-Sun" "Wed" "Thurs" "Tues & Sat" ...
##  $ TIME                     : chr  "Mon-Fri 10am-5pm, Sat-Sun 10am-4pm " "11am-1pm" "3-7pm" "Tues 10a
##  $ MONTHS                   : chr  "Late May - November" "May - November" "May - November " "Tues- Ju
##  $ ACCEPT_SNAP_ACCESS       : chr  "Y" "Applied" NA NA ...
##  $ ACCEPT_FMNP              : chr  "Y" "Applied" "Y" "Y" ...
##  $ ACCEPT_PHILLY_FOOD_BUCKS_: logi  NA NA NA NA NA NA ...
##  $ MAJOR_BUS_SUBWAY_ROUTES  : chr  "25, 39, 43, 89" "3, 39, 89" "15B, 54, 60" "9, 17" ...
```

```
#now all variables containing strings are of class 'character'
```

Change class of variable

What if we want to consider a character variable as a factor? In this dataset, "NEIGHBORHOOD" is the most likely factor variable since it has discernable levels

```
markets$NEIGHBORHOOD <- as.factor(markets$NEIGHBORHOOD)
class(markets$NEIGHBORHOOD) # now it is a factor
```

```
## [1] "factor"
```

Factor variables have levels. Notice that some of these levels are redundant

```
levels(markets$NEIGHBORHOOD)
```

```
##  [1] "Bridesburg Kensington Port Richmond"
##  [2] "Center city"
##  [3] "Center City"
##  [4] "Center City "
##  [5] "Germantown Chestnut Hill"
##  [6] "Lower Northeast"
##  [7] "North"
```

```
##  [8] "Northeast"
##  [9] "Northwest"
## [10] "Northwest "
## [11] "Olney Oak Lane"
## [12] "Roxborough Manayunk"
## [13] "South"
## [14] "Southwest"
## [15] "West"
## [16] "West "
```

Dealing with inconsistent values

White space in variable values

https://bookdown.org/lyzhang10/lzhang_r_tips_book/how-to-deal-with-empty-spaces.html

```
help(str_trim)
```

```
## starting httpd help server ... done
```

```
markets <- markets %>% #note this saves the changes to the dataframe
  mutate(NEIGHBORHOOD = str_trim(NEIGHBORHOOD))
```

```
unique(markets$NEIGHBORHOOD)
```

```
##  [1] "Bridesburg Kensington Port Richmond"
##  [2] "Center City"
##  [3] "Center city"
##  [4] "Germantown Chestnut Hill"
##  [5] "Lower Northeast"
##  [6] "North"
##  [7] "Northeast"
##  [8] "Northwest"
##  [9] "Olney Oak Lane"
## [10] "Roxborough Manayunk"
## [11] "South"
## [12] "Southwest"
## [13] "West"
```

Capitalization in variable values using grep

```
index <- grepl("city", markets$NEIGHBORHOOD, ignore.case = TRUE) #CITY cITY cItY
```

```
markets$NEIGHBORHOOD[index] = "Center City"
```

```
unique(markets$NEIGHBORHOOD)
```

```
##  [1] "Bridesburg Kensington Port Richmond"
##  [2] "Center City"
##  [3] "Germantown Chestnut Hill"
##  [4] "Lower Northeast"
##  [5] "North"
##  [6] "Northeast"
##  [7] "Northwest"
##  [8] "Olney Oak Lane"
##  [9] "Roxborough Manayunk"
## [10] "South"
```

```
## [11] "Southwest"
## [12] "West"
```

Let's check our work

```
levels(markets$NEIGHBORHOOD)
```

```
## NULL
```

Why is this back to being a character? str_trim and grepl operate on character vectors so they likely coerce NEIGHBORHOOD to character

```
class(markets$NEIGHBORHOOD)
```

```
## [1] "character"
```

```
markets$NEIGHBORHOOD <- as.factor(markets$NEIGHBORHOOD) #now we are back to meaningful levels
```

Reshaping / pivoting data

Right now our data is in 'long' format, if we wanted to organize by NEIGHBORHOOD and put the data in 'wide' format, we could use reshape() in Base R:

```
markets_wide <- reshape(markets, idvar = "NEIGHBORHOOD", timevar = "OBJECTID", direction = "wide")
View(markets_wide)
```

To pivot data, we can use the reshape package and the melt and cast functions

Saving out your cleaned dataset

This is useful so you don't need to repeat each of these steps for future analysis! Save the script you clean your data in, and save a copy of the raw data for reference

```
write.csv(markets, "cleaned_Farmers_Markets.csv")
```