Code ▾

Castellano_CS636_Lab02

February 3, 2020

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Question 1 Home Depot Data

a) Read in Home Dept Data (all csv files) and find their dimensions.

https://www.kaggle.com/c/home-depot-product-search-relevance/data (https://www.kaggle.com/c/home-depot-product-search-relevance/data)

The following sets up an API to access Kaggle data directly from R.

Run:

> install.packages("devtools)

> devtools::install_github(“mkearney/kaggler”)

Note You may have to install Rtools. If needed, it can be found here:

https://cran.r-project.org/bin/windows/Rtools/ (https://cran.r-project.org/bin/windows/Rtools/)

Hide

```
library(kaggler)
kgl_auth(username = "andrescastellano", key = "2c9485c3957e5c115b8f255ca0e52378")
```

```
Your Kaggle key has been recorded for this session and saved as `KAGGLE_PAT`
  environment variable for future sessions.
```

```
<request>
Options:
* httpauth: 1
* userpwd: andrescastellano:2c9485c3957e5c115b8f255ca0e52378
```

Hide

```
kgl_competitions_list(search="depot")
```

```
Unauthorized (HTTP 401).
```

```
Response [https://www.kaggle.com/api/v1/competitions/list?page=1&search=depot]
  Date: 2020-02-09 19:54
  Status: 401
  Content-Type: <unknown>
<EMPTY BODY>
```

Competition Id is 4853

Hide

```
c1_datalist <- kgl_competitions_data_list(4853)
```

```
Unauthorized (HTTP 401).
```

Hide

```
c1_datalist
```

```
Response [https://www.kaggle.com/api/v1/competitions/data/list/4853]
  Date: 2020-02-09 19:54
  Status: 401
  Content-Type: <unknown>
<EMPTY BODY>
```

Hide

```
c1_data <- kgl_competitions_data_download(4853,c1_datalist$attributes.csv.zip)
```

```
Internal Server Error (HTTP 500).
```

The API is not working. Will try and fix later.

Loading all .csv files

Hide

```
attributes <- read.csv('C:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/attributes.csv/attributes.csv') # Attributes

dim(attributes)
```

```
[1] 2044803       3
```

Hide

```
prod_desc <- read.csv('C:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/product_descriptions.csv/product_descriptions.csv')

dim(prod_desc)
```

```
[1] 124428      2
```

Hide

```
test <- read.csv('C:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/test.csv/test.csv')

dim(test)
```

```
[1] 166693      4
```

Hide

```
train <- read.csv('c:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/train.csv/train.csv')

dim(train)
```

```
[1] 74067     5
```

b) Show the right down corner element of each file in R.

Hide

```
attributes[2044803,3]
```

```
[1] Power Tool
307591 Levels:  'U.S Patented' 'U.S. Patented' ... ZZZ 234 M08 is designed specificall
y to work with VELUX FS M08, VS M08, VSE M08 and VSS M08 deck mount skylight models
```

Hide

```
prod_desc[124428,2]
```

```
[1] The Bosch quick change bi-metal hole saws feature Progressor tooth geometry, combi
ning cutting teeth with specially designed chip-removal teeth for super-fast cutting a
ction in metal and wood. They work with mandrel models HSBAM, PCM38, PCM12 and PCMSDSP
L. Hole saws 1-1/2 in. and larger can also be used with mandrel model HSBAMP.Progresso
r tooth design for faster cutting and longer lifeReinforced shoulder for increased str
ength8% Cobalt alloy has higher heat resistance10-degree cutting angle for high perfor
mance
110128 Levels: "Building Outdoor Structures" offers practical, easy-to-follow instruct
ions on enhancing any home's front and backyard with the natural beauty of wood. Start
ing with the simple uses of wood in landscaping, such as raised beds, author Scott McB
ride shows the average DIYer how to build retaining walls, arbors, pergolas and 7 othe
r projects, including a gazebo. The book covers everything from choosing materials to
building techniques.House and home-outdoor and recreational areas generalHouse and hom
e-do-it-yourself carpentryGarden structuresDesign and construction ...
```

Hide

```
test[nrow(test),ncol(test)]
```

```
[1] 4 inch hole saw
22427 Levels: '1-3/4' tap wrench ...
```

Hide

```
train[nrow(train),ncol(train)]
```

```
[1] 2.33
```

c) Output the odd numbers of columns and even number of rows of train.csv

Hide

```
train[c(FALSE,TRUE),c(TRUE,FALSE)]
```

	id <int>
2	3
4	16
6	18
8	21
10	27
12	35
14	38
16	51
18	69
20	81

1-10 of 37,033 rows | 1-2 of 3 columns   Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
NA
```

d) Save into R objects and load them using dput, dget, save, load, save.image.

Hide

```
fil <- tempfile()
c <- train[c(FALSE,TRUE),c(TRUE,FALSE)]
# dput(c)
```

e) Install teh Readr pakage from CRAN.

f) Any difference in terms of speed and loading the data? Write a simple code to print out the time cost of reading the test.csv. data using read.csv or read_csv.

Using *read_csv*

Hide

```
library(readr)
system.time(read_csv('C:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/test.c
sv/test.csv'))
```

```
Parsed with column specification:
cols(
  id = ⍰[32mcol_double()⍰[39m,
  product_uid = ⍰[32mcol_double()⍰[39m,
  product_title = ⍰[31mcol_character()⍰[39m,
  search_term = ⍰[31mcol_character()⍰[39m
)
```

```
   user  system elapsed 
   0.42    0.04    0.55 
```

Using *read.csv*

Hide

```
system.time(read.csv('C:/Users/Castellano/Documents/Spring2020/CS636/Home Depot/test.c
sv/test.csv'))
```

```
   user  system elapsed 
   4.36    0.08    4.45 
```

Question 2

a) Create a new vector called “test” containing five numbers of your choice.

Hide

```
test <- c(1,2,3,4,5)
```

b) Create a second vector called “students” containing five common names.

Hide

```
students <- c('Michelle','Bowie','Juan','Andres','James')
```

c) Determine the class of test and students

Hide

```
class(test)
```

```
[1] "numeric"
```

Hide

```
class(students)
```

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

d) Create a data frame containing two columns students and test as defined above.

Hide

```
dat <- data.frame(cbind(students,test))
class(dat)
```

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

e) Convert “test” to character class, and confrim that you were succesful.

Hide

```
test <- as.character(test)
class(test)
```

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

Question 3

a) Select just sepal lenght and species columns from the Iris data set and save the result to a new data.frame named iris2.

Hide

```
data(iris)
iris2 <- data.frame(iris$Sepal.Length,iris$Species)
colnames(iris2) <- c("Sepal Length", "Species")
head(iris2)
```

	Sepal Length <dbl>	**Species** <fctr>
1	5.1	setosa
2	4.9	setosa
3	4.7	setosa
4	4.6	setosa
5	5.0	setosa
6	5.4	setosa

6 rows

Hide

```
NA
```

d) Calculate the mean of the sepal length column in iris2.

Hide

```
avg_sep_length <- mean(iris2$`Sepal Length`)
```

c) Calculate the mean of sepal.length, but only for setosa species

Hide

```
# setosas <- subset(iris2, Species == 'setosa')
# mean(setosas$`Sepal Length`)
mean(subset(iris2, Species == 'setosa')$'Sepal Length')
```

```
[1] 5.006
```

## **d)** Calculate the number of sepal lengths that are more than one standard deviation below the average sepal length

Hide

```
std_dev <- sd(iris2$`Sepal Length`)
Low_Bound <- avg_sep_length - std_dev
nrow(iris2[iris2$'Sepal Length' < Low_Bound,])
```

```
[1] 32
```

Question 4 Write R commands for the following questions:

a) 1000, 1000, 998, 998, 996, 996, …… , 4, 4, 2, 2

Hide

```
rep(seq(from = 1000, to = 2, by = -2), each = 2)
```

```
  [1] 1000 1000  998  998  996  996  994  994  992  992  990  990
 [13]  988  988  986  986  984  984  982  982  980  980  978  978
 [25]  976  976  974  974  972  972  970  970  968  968  966  966
 [37]  964  964  962  962  960  960  958  958  956  956  954  954
 [49]  952  952  950  950  948  948  946  946  944  944  942  942
 [61]  940  940  938  938  936  936  934  934  932  932  930  930
 [73]  928  928  926  926  924  924  922  922  920  920  918  918
 [85]  916  916  914  914  912  912  910  910  908  908  906  906
 [97]  904  904  902  902  900  900  898  898  896  896  894  894
[109]  892  892  890  890  888  888  886  886  884  884  882  882
[121]  880  880  878  878  876  876  874  874  872  872  870  870
[133]  868  868  866  866  864  864  862  862  860  860  858  858
[145]  856  856  854  854  852  852  850  850  848  848  846  846
[157]  844  844  842  842  840  840  838  838  836  836  834  834
[169]  832  832  830  830  828  828  826  826  824  824  822  822
[181]  820  820  818  818  816  816  814  814  812  812  810  810
[193]  808  808  806  806  804  804  802  802  800  800  798  798
[205]  796  796  794  794  792  792  790  790  788  788  786  786
[217]  784  784  782  782  780  780  778  778  776  776  774  774
[229]  772  772  770  770  768  768  766  766  764  764  762  762
[241]  760  760  758  758  756  756  754  754  752  752  750  750
[253]  748  748  746  746  744  744  742  742  740  740  738  738
[265]  736  736  734  734  732  732  730  730  728  728  726  726
[277]  724  724  722  722  720  720  718  718  716  716  714  714
[289]  712  712  710  710  708  708  706  706  704  704  702  702
[301]  700  700  698  698  696  696  694  694  692  692  690  690
[313]  688  688  686  686  684  684  682  682  680  680  678  678
[325]  676  676  674  674  672  672  670  670  668  668  666  666
[337]  664  664  662  662  660  660  658  658  656  656  654  654
[349]  652  652  650  650  648  648  646  646  644  644  642  642
[361]  640  640  638  638  636  636  634  634  632  632  630  630
[373]  628  628  626  626  624  624  622  622  620  620  618  618
[385]  616  616  614  614  612  612  610  610  608  608  606  606
[397]  604  604  602  602  600  600  598  598  596  596  594  594
[409]  592  592  590  590  588  588  586  586  584  584  582  582
[421]  580  580  578  578  576  576  574  574  572  572  570  570
[433]  568  568  566  566  564  564  562  562  560  560  558  558
[445]  556  556  554  554  552  552  550  550  548  548  546  546
[457]  544  544  542  542  540  540  538  538  536  536  534  534
[469]  532  532  530  530  528  528  526  526  524  524  522  522
[481]  520  520  518  518  516  516  514  514  512  512  510  510
[493]  508  508  506  506  504  504  502  502  500  500  498  498
[505]  496  496  494  494  492  492  490  490  488  488  486  486
[517]  484  484  482  482  480  480  478  478  476  476  474  474
[529]  472  472  470  470  468  468  466  466  464  464  462  462
[541]  460  460  458  458  456  456  454  454  452  452  450  450
[553]  448  448  446  446  444  444  442  442  440  440  438  438
[565]  436  436  434  434  432  432  430  430  428  428  426  426
```

```
 [577]  424  424  422  422  420  420  418  418  416  416  414  414
 [589]  412  412  410  410  408  408  406  406  404  404  402  402
 [601]  400  400  398  398  396  396  394  394  392  392  390  390
 [613]  388  388  386  386  384  384  382  382  380  380  378  378
 [625]  376  376  374  374  372  372  370  370  368  368  366  366
 [637]  364  364  362  362  360  360  358  358  356  356  354  354
 [649]  352  352  350  350  348  348  346  346  344  344  342  342
 [661]  340  340  338  338  336  336  334  334  332  332  330  330
 [673]  328  328  326  326  324  324  322  322  320  320  318  318
 [685]  316  316  314  314  312  312  310  310  308  308  306  306
 [697]  304  304  302  302  300  300  298  298  296  296  294  294
 [709]  292  292  290  290  288  288  286  286  284  284  282  282
 [721]  280  280  278  278  276  276  274  274  272  272  270  270
 [733]  268  268  266  266  264  264  262  262  260  260  258  258
 [745]  256  256  254  254  252  252  250  250  248  248  246  246
 [757]  244  244  242  242  240  240  238  238  236  236  234  234
 [769]  232  232  230  230  228  228  226  226  224  224  222  222
 [781]  220  220  218  218  216  216  214  214  212  212  210  210
 [793]  208  208  206  206  204  204  202  202  200  200  198  198
 [805]  196  196  194  194  192  192  190  190  188  188  186  186
 [817]  184  184  182  182  180  180  178  178  176  176  174  174
 [829]  172  172  170  170  168  168  166  166  164  164  162  162
 [841]  160  160  158  158  156  156  154  154  152  152  150  150
 [853]  148  148  146  146  144  144  142  142  140  140  138  138
 [865]  136  136  134  134  132  132  130  130  128  128  126  126
 [877]  124  124  122  122  120  120  118  118  116  116  114  114
 [889]  112  112  110  110  108  108  106  106  104  104  102  102
 [901]  100  100   98   98   96   96   94   94   92   92   90   90
 [913]   88   88   86   86   84   84   82   82   80   80   78   78
 [925]   76   76   74   74   72   72   70   70   68   68   66   66
 [937]   64   64   62   62   60   60   58   58   56   56   54   54
 [949]   52   52   50   50   48   48   46   46   44   44   42   42
 [961]   40   40   38   38   36   36   34   34   32   32   30   30
 [973]   28   28   26   26   24   24   22   22   20   20   18   18
 [985]   16   16   14   14   12   12   10   10    8    8    6    6
 [997]    4    4    2    2
```

b) Generate a sequence of 10 “a” and 5 “b”

Hide

```
rep(c('a','b'), c(10,5))
```

```
 [1] "a" "a" "a" "a" "a" "a" "a" "a" "a" "a" "b" "b" "b" "b" "b"
```

c) Print rever the order of **b)**

Hide

```
rev(rep(c('a','b'),c(10,5)))
```

```
 [1] "b" "b" "b" "b" "b" "a" "a" "a" "a" "a" "a" "a" "a" "a" "a"
```

Question 5

Find the row numbers in the iris data set, where the Petal.Length is larger than 5 and Petal.Width is less than 1.7. And print out this part of the iris data set.

Hide

```
iris[iris$Petal.Length > 5 & iris$Petal.Width < 1.7,]
```

	Sepal.Length <dbl>	**Sepal.Width** <dbl>	**Petal.Length** <dbl>	**Petal.Width** <dbl>	**Species** <fctr>
84	6.0	2.7	5.1	1.6	versicolor
130	7.2	3.0	5.8	1.6	virginica
134	6.3	2.8	5.1	1.5	virginica
135	6.1	2.6	5.6	1.4	virginica

4 rows

Question 6

Guess what the following matrix would look like and the results of the following commands and compare with the real results.

x <- matrix(c(rep(6,3), seq(10,2,-3),x(NA,3,4),6,1,10),4,3)

Hide

```
x <- matrix(c(rep(6,3), seq(10,2,-3),c(NA,3,4), 6,1,10), 4, 3)
```

print(x[,x[2,] > 4])

Select from matrix x, all the rows and columns for which the second row of any column is greater than 4.

Hide

```
print(x[,x[2,] > 4])
```

```
     [,1] [,2]
[1,]    6    4
[2,]    6    6
[3,]    6    1
[4,]   10   10
```

print(x[,2] < 4)

Print the elements of x for which the second column is less than 4

Hide

```
print(x[,2] < 4)
```

```
[1] FALSE FALSE    NA  TRUE
```

Wrong, this code prints logical whether or not the elements are < 4

print(x[x[,2] < 4,])

Prints the actual values of x for which the second column is less than 4

Hide

```
print(x[x[,2] < 4,])
```

```
     [,1] [,2] [,3]
[1,]   NA   NA   NA
[2,]   10    3   10
```

Don’t actually understand what this did.

sum(x[x > 6]) Sums all the values of elements of x > 6

Hide

```
x
```

```
     [,1] [,2] [,3]
[1,]    6    7    4
[2,]    6    4    6
[3,]    6   NA    1
[4,]   10    3   10
```

Hide

```
sum(x[x > 6])
```

```
[1] NA
```

Don’t understand NA.

sum(x[x > 6],na.rm=T)

This shall remove the NAs from Calc. Which makes me think the reason it didnt work before, is because you cannot add numbers to NAs.

Hide

```
sum(x[x > 6],na.rm=T)
```

```
[1] 27
```

order(x[,3]) This should order the elements of x along the third axis in ascending order.

Hide

```
order(x[,3])
```

```
[1] 3 1 2 4
```

It didn’t. It ordered the INDICES of the matrix according to the increasing value of elements.

x[order(x[,3]),]

This should do what I thought the prvious code was going to do.

Hide

```
x[order(x[,3]),]
```

```
     [,1] [,2] [,3]
[1,]    6   NA    1
[2,]    6    7    4
[3,]    6    4    6
[4,]   10    3   10
```

It did.

Thanks.