More Exploration in Data Manipulation

Yutong Song
10/29/2017

Introduction

During lab05 we have learned about a set of five basic verbs in dplyr package to manipulate a data frame. I was amazed by how a single function can give a data frame a whole new look. So I decide to explore some more interesting way to manipulate a data frame with two packages: this first one is the dplyr package we already downloaded, and the other is called "tibble package", which is another very useful, simple package for data manipulation. I will explore the data frame manipulation in four main aspects:

1. Scrutinizing the data
2. Changing or adding columns
3. Manipulating cases
4. Names of rows

Package Installation and Data Preparation

I assume that you already installed the dplyr package on the computer, and if that is not the case, you can run the following codes in your console:

```
install.packages("dplyr")
```

Also, to download the tibble package:(run the following codes in your console)

```
install.packages("tibble")
```

And then to use the dplyr and tibble package, we run the following codes:

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tibble)
```

For the next step, to visualize more functions in dplyr and tibble package, I am going to use the data frame `nba2017-players.csv` to illustrate some of the functions. Since we have used the `nba2017-players.csv` before, we can import the data:

```
dat <- read.csv('nba2017-players.csv', stringsAsFactors = FALSE)
```

If you deleted the data, you can download the data again using the following codes:

```
github <- "https://github.com/ucb-stat133/stat133-fall-2017/raw/master/"
csv <- "data/nba2017-players.csv"
download.file(url = paste0(github, csv), destfile = 'nba2017-players.csv')
```

Scrutinizing the data

> ## Functions:
>
> - `first()`
> - `last()`
> - `nth()`

`first()`, `last()`, and `nth()` enable you to scrutinize the first, last, or nth column of a data frame or the first, last, or nth value of a specific column.

```
# to look at the the first 5 values of the first column
head(first(dat), 5)
```

```
## [1] "Al Horford"        "Amir Johnson"      "Avery Bradley"
## [4] "Demetrius Jackson" "Gerald Green"
```

```
# to look at the first value of the age column
first(dat$age)
```

```
## [1] 30
```

```
# to look at the the first 5 values of the last column
head(last(dat), 5)
```

```
## [1] 108  67  68   3  33
```

```
# to look at the last value of the games column
last(dat$games)
```

```
## [1] 47
```

```
# to look at the the first 5 values of the 4th column
head(nth(dat, 4), 5)
```

```
## [1] 82 81 74 73 79
```

```
# to look at the 2nd value of the points3 column
nth(dat$points3, 2)
```

```
## [1] 27
```

Changing or adding columns

> ### Functions:
>
> - `rename()`
> - `transmute()`
> - `mutate_at()`
> - `mutate_if()`

`rename()` allows you to change names of each column. Suppose I want to rename the `salary` column as `payroll`:

```
# rename salary column as payroll
dat <- rename(dat, payroll = salary)

# the name has been changed to payroll
names(dat)
```

```
##  [1] "player"     "team"       "position"   "height"     "weight"
##  [6] "age"        "experience" "college"    "payroll"    "games"
## [11] "minutes"    "points"     "points3"    "points2"    "points1"
```

If the player's minutes per game as the only column I want to keep, we can use the `transmute()`:

```
# create a new column mins_per_game and only keep this column and drop the other columns
dat_mins_per_game_only <- transmute(dat, mins_per_game = minutes / games)
# we can see in the newly created data frame dat_mins_per_game_only, mins_per_game is the only column left.
```

Moreover, suppose we want to apply operations to different columns simultaneously, we can use `mutate_at()` to manually select the columns we want to apply the operation, or we can use `mutate_if()` to automatically select the columns using the condition we set inside the function.

For example, to begin with, we first create a data frame `cavs` of Cavalier players with their `points3`, `points2`, and `points1`:

```
# create data frame cavs
cavs <- select(filter(dat, team == "CLE"), player, points3, points2, points1)
```

Then, suppose we want to change 3 columns to `cavs` by applying log transformation to `points3`, `points2` and `points1` for graphical analysis, one way to do this is `mutate_at()`:

```
cavs_log <- mutate_at(cavs, .vars = vars(points3, points2, points1), funs(log(.)))
# we manually specify columns in this function
cavs_log
```

```
##              player  points3  points2  points1
## 1      Channing Frye 4.919981 4.615121 4.143135
## 2      Dahntay Jones     -Inf 1.098612 1.098612
## 3     Deron Williams 3.091042 3.828641 3.044522
## 4   Derrick Williams 3.044522 3.496508 3.295837
## 5        Edy Tavares     -Inf 1.098612     -Inf
## 6      Iman Shumpert 4.543295 4.672829 4.262680
## 7         J.R. Smith 4.553877 3.332205 2.302585
## 8        James Jones 3.433987 2.564949 2.564949
## 9         Kay Felder 1.945910 4.007333 3.555348
## 10        Kevin Love 4.976734 5.416100 5.549076
## 11       Kyle Korver 4.574711 3.526361 2.639057
## 12      Kyrie Irving 5.176150 6.202536 5.693732
## 13      LeBron James 4.820282 6.416732 5.880533
## 14 Richard Jefferson 4.127134 4.510860 4.382027
## 15  Tristan Thompson     -Inf 5.568345 4.663439
```

And we can also achieve so by using `mutate_if()`:

```
cavs_log_2ndmethod <- mutate_if(cavs, is.numeric, funs(log(.)))
# the condition is.numeric automatically finds the points3, points2, and points1 columns
cavs_log_2ndmethod
```

```
##              player  points3  points2  points1
## 1      Channing Frye 4.919981 4.615121 4.143135
## 2      Dahntay Jones     -Inf 1.098612 1.098612
## 3     Deron Williams 3.091042 3.828641 3.044522
## 4   Derrick Williams 3.044522 3.496508 3.295837
## 5        Edy Tavares     -Inf 1.098612     -Inf
## 6      Iman Shumpert 4.543295 4.672829 4.262680
## 7         J.R. Smith 4.553877 3.332205 2.302585
## 8        James Jones 3.433987 2.564949 2.564949
## 9         Kay Felder 1.945910 4.007333 3.555348
## 10        Kevin Love 4.976734 5.416100 5.549076
## 11       Kyle Korver 4.574711 3.526361 2.639057
## 12      Kyrie Irving 5.176150 6.202536 5.693732
## 13      LeBron James 4.820282 6.416732 5.880533
## 14 Richard Jefferson 4.127134 4.510860 4.382027
## 15  Tristan Thompson     -Inf 5.568345 4.663439
```

Manipulating cases

> ## Functions:
>
> - `add_row()`
> - `distinct()`
> - `sample_n()`
> - `sample_frac()`

`add_row()` can add a row to the data frame:

```
# create new player Sunny
dat_sunny <- add_row(dat, player = "Sunny", team = "CLE", position = "C", height = 80, weight = 199, age =
tail(dat_sunny, 1) # by default the newly created row is at the bottom of the data frame
```

```
##     player team position height weight age experience
## 442  Sunny  CLE        C     80    199  20          0
##                             college payroll games minutes points points3
## 442 University of California Berkeley       1     0       0      0       0
##     points2 points1
## 442       0       0
```

We can also specify the position where we want the new player in the list by using parameters .before and .afterIf a new player, Oski, joins the NBA league this year, we can use `add_row()` to list him in our data:

```
# in this case lets put a player Oski in the first row
dat_oski <- add_row(dat, player = "Oski", team = "CLE", position = "C", height = 80, weight = 199, age = 20
head(dat_oski, 1) # Oski now is the first case in this data frame
```

```
##   player team position height weight age experience
## 1   Oski  CLE        C     80    199  20          0
##                           college payroll games minutes points points3
## 1 University of California Berkeley       1     0       0      0       0
##   points2 points1
## 1       0       0
```

Suppose in a data analysis we want to study the spread of 3 points, and in this case we may want to extract the rows with duplicate values of `points3`, and `distinct()` can achieve that:

```
# remove cases with duplicate value of 3 points
dat_noduplicate <- distinct(dat, points3)
nrow(dat_noduplicate) #we can observe that 295 of the cases have beem removed!
```

```
## [1] 147
```

Moreover, many of the quantitative analysis need to draw random samples from a data frame, and this when `sample_n()` and `sample_frac()` come in use, to draw random samples by either numbers or fractions:

```
# to randomly draw 5 samples from dat
sample_n(dat, 5)
```

```
##                   player team position height weight age experience
## 175    Willy Hernangomez  NYK        C     83    240  22          0
## 213      Justin Hamilton  BRK        C     84    260  26          2
## 72       Thabo Sefolosha  ATL       SF     79    220  32         10
## 415          David Nwaba  LAL       SG     76    209  24          0
## 389 Georgios Papagiannis  SAC        C     85    240  19          0
##                                                     college payroll games
## 175                                                        1375000    72
## 213                                Louisiana State University 3000000    64
## 72                                                         3850000    62
## 415 California Polytechnic State University, San Luis Obispo   73528    20
## 389                                                        2202240    22
##     minutes points points3 points2 points1
## 175    1324    587       4     242      91
## 213    1177    442      55     119      39
## 72     1596    444      41     133      55
## 415     397    120       1      46      25
## 389     355    124       0      56      12
```

```
# to random draw samples from dat, and the number of samples equals 2% of the total cases
sample_frac(dat, 0.02)
```

```
##                 player team position height weight age experience
## 255       Isaiah Taylor  HOU       PG     75    170  22          0
## 56          Otto Porter  WAS       SF     80    198  23          3
## 185 Marcus Georges-Hunt  ORL       SG     77    216  22          0
## 23          James Jones  CLE       SF     80    218  36         13
## 307        Steven Adams  OKC        C     84    255  23          3
## 256        James Harden  HOU       PG     77    220  27          7
## 419     Larry Nance Jr.  LAL       PF     81    230  24          1
## 116         Robin Lopez  CHI        C     84    255  28          8
## 94       Kevin Seraphin  IND       PF     81    285  27          6
##                            college  payroll games minutes points points3
## 255   University of Texas at Austin   255000     4      52      3       0
## 56            Georgetown University  5893981    80    2605   1075     148
## 185 Georgia Institute of Technology    31969     5      48     14       1
## 23              University of Miami  1551659    48     381    132      31
## 307        University of Pittsburgh  3140517    80    2389    905       0
## 256        Arizona State University 26540100    81    2947   2356     262
## 419           University of Wyoming  1207680    63    1442    449      10
## 116             Stanford University 13219250    81    2271    839       0
## 94                                    1800000    49     559    232       0
##     points2 points1
## 255       1       1
## 56      266      99
## 185       1       9
## 23       13      13
## 307     374     157
## 256     412     746
## 419     180      59
## 116     382      75
## 94      109      14
```

```
# if we want to draw samples with replacement, change to parameter replace:
sample_n(dat, 5, replace = TRUE)
```

```
##               player team position height weight age experience
## 158    Nicolas Batum  CHO       SG     80    200  28          8
## 61   Dennis Schroder  ATL       PG     73    172  23          3
## 210 Isaiah Whitehead  BRK       PG     76    213  21          0
## 229     Kevon Looney  GSW        C     81    220  20          1
## 64     Jose Calderon  ATL       PG     75    200  35         11
##                                 college  payroll games minutes points
## 158                                     20869566    77    2617   1164
## 61                                       2708582    79    2485   1414
## 210                 Seton Hall University  1074145    73    1643    543
## 229 University of California, Los Angeles  1182840    53     447    135
## 64                                       392478    17     247     61
##     points3 points2 points1
## 158     135     258     243
## 61      100     448     218
## 210      44     160      91
## 229       2      54      21
## 64        8      15       7
```

```
sample_frac(dat, 0.02, replace = TRUE)
```

```
##               player team position height weight age experience
## 302      Kyle Singler  OKC       SF     80    228  28          4
## 105 Cristiano Felicio  CHI        C     82    275  24          1
## 181     Elfrid Payton  ORL       PG     76    185  22          2
## 384     Arron Afflalo  SAC       SG     77    210  31          9
## 35    DeMarre Carroll  TOR       SF     80    215  30          7
## 288       Joe Johnson  UTA       SF     79    240  35         15
## 161   Carmelo Anthony  NYK       SF     80    240  32         13
## 33       Delon Wright  TOR       PG     77    183  24          1
## 370      Devin Harris  DAL       PG     75    192  33         12
##                                  college  payroll games minutes points
## 302                       Duke University  4837500    32     385     88
## 105                                        874636    66    1040    316
## 181  University of Louisiana at Lafayette  2613600    82    2412   1046
## 384 University of California, Los Angeles 12500000    61    1580    515
## 35                 University of Missouri 14200000    72    1882    638
## 288                University of Arkansas 11000000    78    1843    715
## 161                   Syracuse University 24559380    74    2538   1659
## 33                     University of Utah  1577280    27     446    150
## 370               University of Wisconsin  4228000    65    1087    437
##     points3 points2 points1
## 302       7      27      13
## 105       0     128      60
## 181      40     390     146
## 384      62     123      83
## 35      109     111      89
## 288     106     167      63
## 161     151     451     304
## 33       10      39      42
## 370      58      78     107
```

Names of rows

> ## Functions:
>
> - `rownames_to_column()`
> - `column_to_rownames()`
> - `remove_rownames()`

Sometimes we want to blend the row names into the data frame as a column, and sometimes we want to use one specific column as the row names or just delete the row names. In tibble package there is a series of functions can help us deal with row names.

`rownames_to_column()` can sets the row names into the data frame as a column:

```
# make row names, 1:442, as the first column, and give this column a name, "Number"
dat_names_to_column <- rownames_to_column(dat, var = "Number")
head(dat_names_to_column, 5) # we can see the change by observing the first five rows
```

```
##   Number            player team position height weight age experience
## 1      1        Al Horford  BOS        C     82    245  30          9
## 2      2      Amir Johnson  BOS       PF     81    240  29         11
## 3      3     Avery Bradley  BOS       SG     74    180  26          6
## 4      4 Demetrius Jackson  BOS       PG     73    201  22          0
## 5      5      Gerald Green  BOS       SF     79    205  31          9
##                          college  payroll games minutes points points3
## 1         University of Florida 26540100    68    2193    952      86
## 2                               12000000    80    1608    520      27
## 3 University of Texas at Austin  8269663    55    1835    894     108
## 4      University of Notre Dame  1450000     5      17     10       1
## 5                               1410598    47     538    262      39
##   points2 points1
## 1     293     108
## 2     186      67
## 3     251      68
## 4       2       3
## 5      56      33
```

`column_to_rownames()`, as you can guess according to the name, can sets one specific column as names of rows:

```r
# use the names of player as the row names
dat_player_as_names <- column_to_rownames(dat, var = "player")
head(dat_player_as_names, 5) # we can see the change by observing the first five rows
```

```
##                   team position height weight age experience
## Al Horford         BOS        C     82    245  30          9
## Amir Johnson       BOS       PF     81    240  29         11
## Avery Bradley      BOS       SG     74    180  26          6
## Demetrius Jackson  BOS       PG     73    201  22          0
## Gerald Green       BOS       SF     79    205  31          9
##                                      college  payroll games minutes
## Al Horford                University of Florida 26540100    68    2193
## Amir Johnson                                    12000000    80    1608
## Avery Bradley     University of Texas at Austin  8269663    55    1835
## Demetrius Jackson      University of Notre Dame  1450000     5      17
## Gerald Green                                    1410598    47     538
##                   points points3 points2 points1
## Al Horford           952      86     293     108
## Amir Johnson         520      27     186      67
## Avery Bradley        894     108     251      68
## Demetrius Jackson     10       1       2       3
## Gerald Green         262      39      56      33
```

Notice that we can not use duplicate values as row names. So if you try to run `column_to_rownames(dat, var = "payroll")`, it will show a error because there is duplicate values in their payrolls.

`remove_rownames()` removes the row names:

```r
# remove all row names
dat_no_names <- remove_rownames(dat_player_as_names)
head(dat_no_names, 5) # we can see the row names, which was player names initially, have been removed
```

```
##   team position height weight age experience                      college
## 1  BOS        C     82    245  30          9         University of Florida
## 2  BOS       PF     81    240  29         11
## 3  BOS       SG     74    180  26          6 University of Texas at Austin
## 4  BOS       PG     73    201  22          0      University of Notre Dame
## 5  BOS       SF     79    205  31          9
##    payroll games minutes points points3 points2 points1
## 1 26540100    68    2193    952      86     293     108
## 2 12000000    80    1608    520      27     186      67
## 3  8269663    55    1835    894     108     251      68
## 4  1450000     5      17     10       1       2       3
## 5  1410598    47     538    262      39      56      33
```

References:

- Data Transformation with dplyr : : CHEAT SHEET
- https://stackoverflow.com/questions/42052078/correct-syntax-for-mutate-if
- https://www.rdocumentation.org/packages/tibble/versions/1.3.4/topics/rownames
- https://www.youtube.com/watch?v=9C0DOyYeipY
- https://stackoverflow.com/questions/29511215/convert-row-names-into-first-column
- https://www.youtube.com/watch?v=0bcA3-6fKDk
- http://dplyr.tidyverse.org/reference/summarise_all.html
- https://www.rdocumentation.org/packages/Momocs/versions/1.1.6/topics/sample_frac