Warmup 04 - Data Wrangling and Visualization

Stat 133, Spring 2019

Introduction

The purpose of this assignment is twofold. On one hand, we want you to keep working with data frames and producing plots but now using the packages `"dplyr"` and `"ggplot2"`. On the other hand, you will have the opportunity to perform aggregation operations (i.e. *group by*), ranking tables, and making various visual displays.

Regarding the Research Question, in this assignment we will focus on the relationship between Salary and Points, but this time as seen from the teams point of view (aggregated data by team).

General Instructions

- Write your narrative and code in an `Rmd` (R markdown) file.
- Name this file as `warmup04-first-last.Rmd`, where `first` and `last` are your first and last names (e.g. `warmup04-gaston-sanchez.Rmd`).
- Please do not use code chunk options such as: `echo = FALSE`, `eval = FALSE`, `results = 'hide'`. All chunks must be visible and evaluated.
- Submit your Rmd and html files to bCourses.

---

1) Importing Data

The first task involves reading the data into R. In the previous assignment, warmup03, you learned how to import tables with base R functions, e.g. `read.table()` and `read.csv()`. Now you will use functions from the R package `"readr"`.

- The data file is located in the github repository for homework assignments (see folder `data/`):

https://github.com/ucb-stat133/stat133-hws/raw/master/data/nba2018-players.csv

- Read the documentation for the `"readr"` function `read_csv()` and examine the parameters that are used to specify whether the data have a header for column names, the data types of each column, encoding of missing values, etc.

- The code below creates a string `datafile` with the url for the location of `nba2018-players.csv`. You can use this string as the value of the argument `file` for `read.table()` and `read.csv()`.

```r
# assembling url so it fits on the screen
github <- 'https://raw.githubusercontent.com/ucb-stat133/stat133-hws/'
repo <- 'master/data/nba2018-players.csv'
datafile <- paste0(github, repo)
```

a) In order to import the data table, you will have to specify data types for each column:

   - Variables `player`, `team`, and `college` must be imported as `"character"` (not as factors).
   - Variable `position` must be imported as `"factor"`.
   - Numeric variables `height`, `weight`, `age`, `experience`, `games`, `minutes`, `points`, `points3`, `points2`, and `points1` must be imported as `"integer"`.
   - Variable `salary` must be imported as `"double"` or `"real"`.

b) Import the data with `read_csv()` and assign it to an object named `dat`. And then display a `summary()` of `dat`.

c) What class of object is `dat`? Run an R command to answer this question.

---

2) Technical Questions about "readr"

a) When you print `dat`—typing the name of the object to see how it gets displayed—R tells you that it is a `tibble`. Do some research (e.g. google it) to explain what are the similarities and differences between a `data.frame` and a `tibble`.

b) Say you want to import just a couple of columns from `nba2018-players.csv`. For instance, suppose you want to import columns `player`, `team`, `salary`, and `points`, with data types `"character"`, `"character"`, `"double"`, and `"double"`, respectively. Is it possible to use `read_csv()` to import only these columns, with the specified data types? Yes or No, and explain.

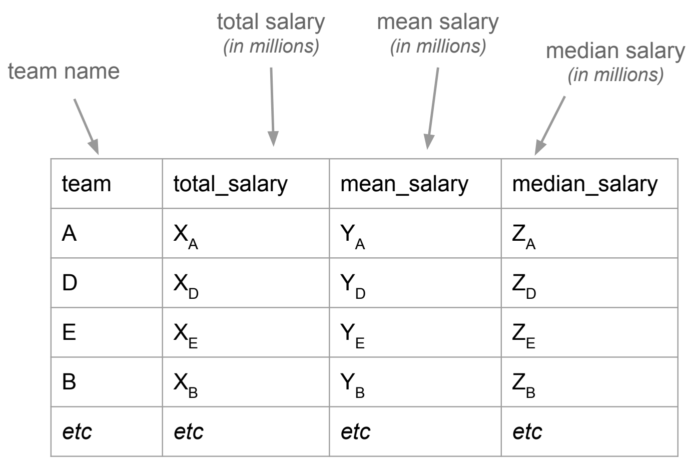
c) The base R function `read.csv()` uses arguments like `header`, `col.names`, `na.strings`, and `colClasses`. What are the equivalent (or similar) arguments in `read_csv()`?

---

3) Salaries by Team

After importing the data, the next task involves obtaining aggregated data of NBA teams in terms of salaries.

a) Use "`dplyr`" functions to create a data `tibble` named `team_salaries` containing columns for `team`, `total_salary`, `mean_salary`, and `median_salary`. All three types of salary variables must be in millions of dollars (NOT in dollars). Arrange this table by `total_salary` in **descending** order.


team name
total salary
(in millions)
mean salary
(in millions)
median salary
(in millions)

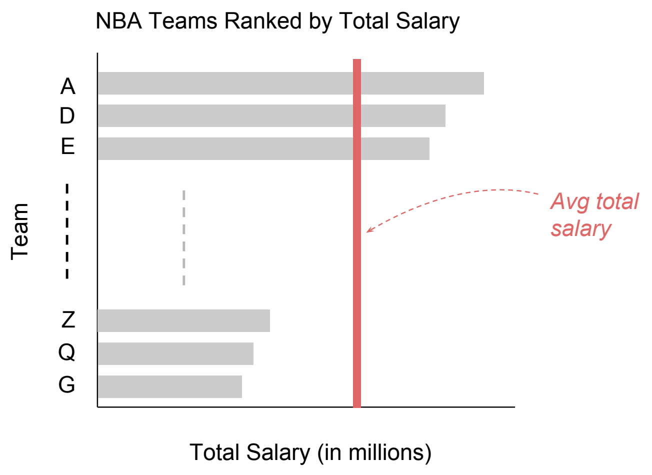
team	total_salary	mean_salary	median_salary
A	X_A	Y_A	Z_A
D	X_D	Y_D	Z_D
E	X_E	Y_E	Z_E
B	X_B	Y_B	Z_B
etc	*etc*	*etc*	*etc*

Values arranged by total salary (in descending order)

b) Because `team_salaries` is a `tibble` object, when you try to print it, only the first 10 rows are displayed. In order to display the entire table, run the following command:

```r
as.data.frame(team_salaries)
```

c) Use "`ggplot2`" functions to create a horizontal barchart of the total salary by team, in decreasing order (see conceptual sketch below). Include axis labels, and a title. Also, notice the vertical line indicating the average salary of NBA teams.



Take a look at the following resources to learn how to obtain such graphic with ggplot.

- Horizontal barplot in ggplot2, and axis labels in ggplot2 https://stackoverflow.com/questions/10941225/horizontal-barplot-in-ggplot2

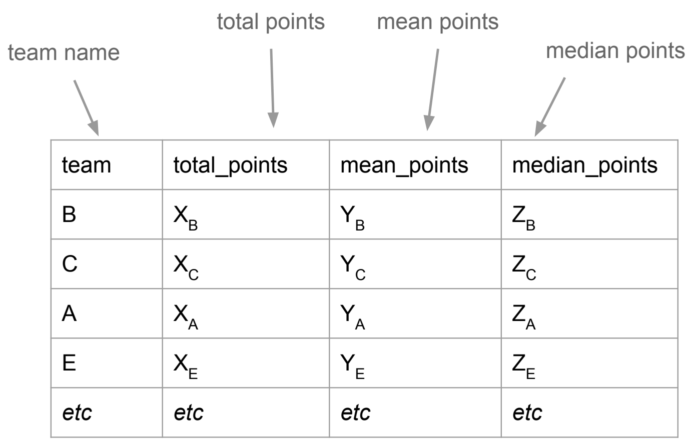
http://ggplot2.tidyverse.org/reference/labs.html

---

4) Points by Team

The fourth task involves obtaining aggregated data of NBA teams in terms of scored points (i.e. total points).

a) Use "`dplyr`" functions to create a data `tibble` named `team_points` containing columns for `team`, `total_points`, `mean_points`, and `median_points`. Arrange this table by `total_points` in descending order.

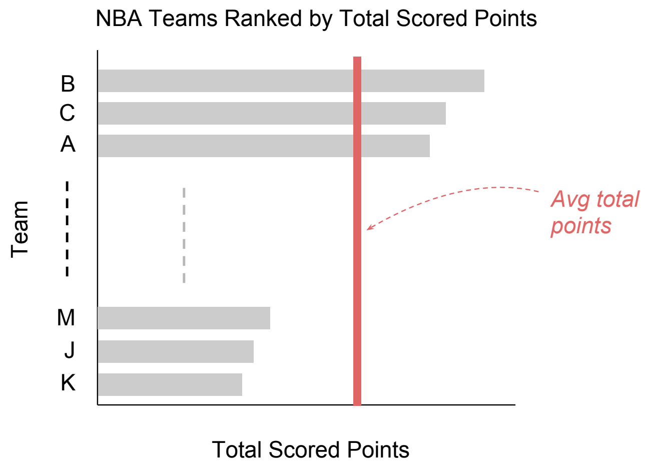

team name
total points
mean points
median points

team	total_points	mean_points	median_points
B	X_B	Y_B	Z_B
C	X_C	Y_C	Z_C
A	X_A	Y_A	Z_A
E	X_E	Y_E	Z_E
etc	*etc*	*etc*	*etc*

Values arranged by total points (in descending order)

b) Invoke `as.data.frame(team_points)` to display the entire table.

c) Use "`ggplot2`" functions to create a barchart of the total points by team, similar to one about salaries in the previous section.



5) Cost of Scored Points

Let's now dive into the association of total-salary and total-points. One way to start studying the relationship between `total_salary` and `total_points` would be by looking at the data with a scatterplot, for instance.

An interesting alternative is to think of the question: *What is the team's cost of a scored point (in terms of its payroll)?* For example, say the total salary of a given team is 100 million dollars, and that the total scored points is 10,000. Then, the "cost" (or the "price") of a scored point for this team would be $\$10,000 = \$100,000,000/10,000$.

a) Join (i.e. merge) your tables `team_salaries` and `team_points` into a single table named `points_salary`. There are several functions in `"dplyr"` that allow you to *join* two tables: e.g. `inner_join()`, `full_join()`, etc. You may want to see some examples:

- https://dplyr.tidyverse.org/reference/join.html
- https://stat545.com/bit001_dplyr-cheatsheet.html

b) Display a `summary()` of `points_salary`

c) Mutate `points_salary` by adding a column `cost_point`; this new variable should be obtained as the ratio of Total-Salary (in dollars) divided by Total-Points.

d) Display a `summary()` of `cost_point`

e) Use `"ggplot2"` functions to create a visual display of `cost_point` values. Use a graphic that allows any reader to easily see: differences between the costs, which team has the highest cost, which team has the lowest cost, and which team(s) has the "typical" cost of points. Keep in mind supporting graphical elements such as axis labels, title, colors, background, maybe a legend, etc.

f) Use `"ggplot2"` functions to create a visual display of salary-values and point-values, taking into account `cost_point`. In other words, choose one salary variable (e.g. `median_salary`), and choose one of the points variable (e.g. `median_points`), to create a visual display that also takes into account `cost_point` as a visual attribute.