**Data Wrangling Final Project**

Yitao Liu

2020/5/3

**Introduction:**

The National Basketball Association (NBA) is a men’s professional basketball league is North America. I like watching NBA since I was a child, so the first idea that comes to my mind is to do something about NBA. Not only because I like NBA, one of the biggest advantages that NBA has is that it provides sufficient and useful data for me to analyze, and the data is not difficult to find and access.

With sufficient data, I find that athlete salaries compared to their stats is an interesting topic for me. So, I decided to wrangle two kinds of data: **The salary data** and **the player stats**, and I want to see if there is any pattern between them. The goal of this project is to scrap the data from different website, wrangle and clean the data, and to see if there’s any connection between them.

**Data Resources:**

There are many different NBA data resources that are available online, but they are all stored in different way. So, I have to choose my resource carefully. After making my goal, there are two kinds of data that are required in my project: **the salary data** and **the** **stats data**. I will talk about the resources of these two data separately.

1. *Stats Data*

After doing some research, I find that **NBA official site** (<https://stats.nba.com/>) provides very detailed players stats. So, this will be my resource of players stats data.

1. *Salary Data*

For salary data, I find that there’s a well scraped data of salary from **Kaggle** website (<https://www.kaggle.com/hultm28/nba-player-salary-data-2002-2017>). This dataset includes NBA player salary data between the 2002/03 and 2017/18 seasons - includes player, team, position, and salary amount. This will be one of my resource of players salary data.

However, the data from Kaggle has not being updated for two years. Therefore, I need to find another resource to get the salary data of 2018/19 and 2019/2020 season. The first resource that I find is through **Basketball-Reference** Website (<https://www.basketball-reference.com/contracts/players.html>). After I scrap the table from the website, I find that there is a problem about this resource: the names of the players do not integrate with the stats data that I found. For example, the following is the name of player Nikola Vucevic on Basketball-Reference:

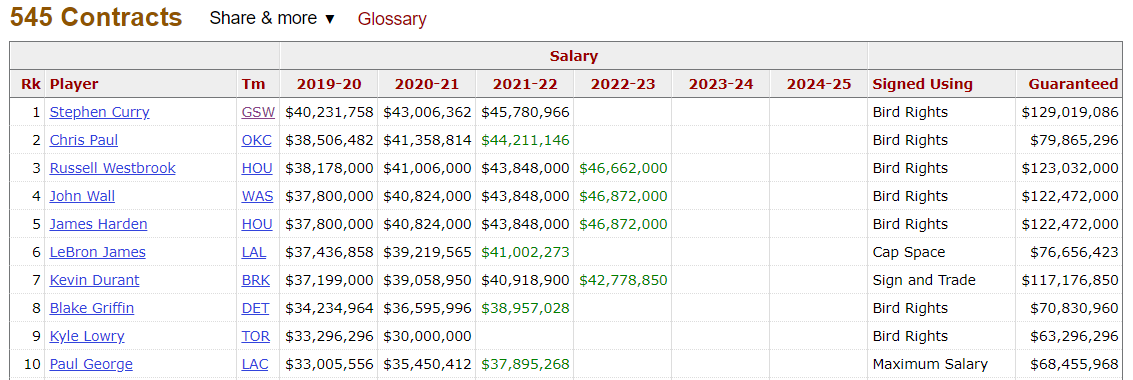


You can notice that the name contains some letters from other language, where the stats data that I found before is only in English:



Therefore, Basketball-Reference is not a good resource for my salary data.

Nevertheless, Basketball-Referenceis not totally useless for this project. Later, I find that there is a contract table on it (<https://www.basketball-reference.com/contracts/players.html>) that could be helpful for making prediction:

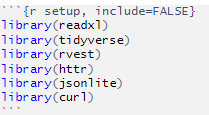


So, I also decide to scrap this table as the resource of players future salary table. However, building statistical model is not the focus of my project, so now I just scrap it for practice.

Next, I find that **ESPN** website (<http://www.espn.com/nba/salaries>) provides perfect salary data for me that the names of the players well match my stats data. Finally, I decide to choose ESPN website as another resource of player salary data.

**Data Collecting:**

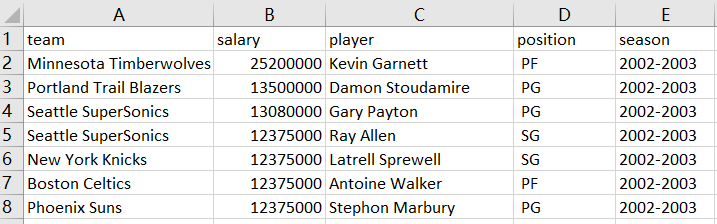
Now it is time to actually collect the data using R code. The code for data scraping is saved in  . First, load some packages that are needed for this project:



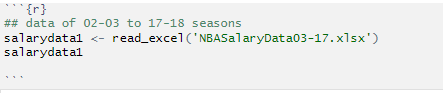
Like above, I will talk about the collecting process of **salary data** and **stats data** separately.

1. *Salary Data*

Firstly, I will collect the player salary data between the 2002/03 and 2017/18 seasons, from **Kaggle**. The data is store in xlsx form: 



Next, read the excel file into R:



The table looks like this, there are 6255 rows:



The collecting of salary data between the 2002/03 and 2017/18 seasons is done.

Next step is to scrap the salary data of 2018/19 and 2019/2020 season from **ESPN**. Because some URL address and API-keys are needed from now on, I create a R file that contains those strings: 

Let’s take a look at the data on ESPN. For each season, there are many tables being on different pages. The URL addresses of 18/19 season are written like this:

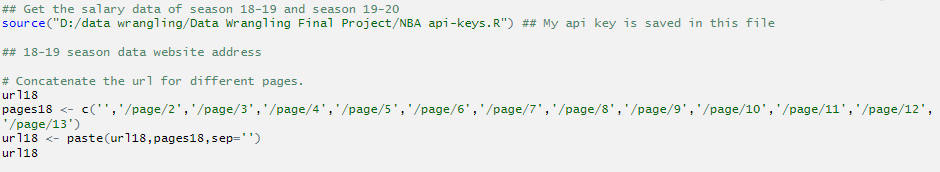
http://www.espn.com/nba/salaries/\_/year/2019

http://www.espn.com/nba/salaries/\_/year/2019/page/2

http://www.espn.com/nba/salaries/\_/year/2019/page/3

…

So, there is a pattern in the addresses for different pages. Therefore, I implement a code using a vector for different pages and concatenate them with the common part:





Do the same thing on 19/20 season address:





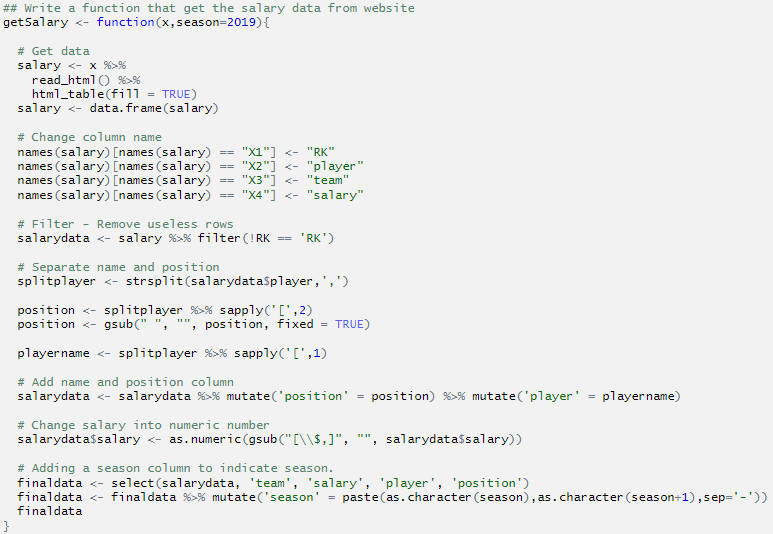
Next, write a function **getSalary()** that read these addresses. Let’s take a look on the website:



For this table, there are several things that must be included in the function:

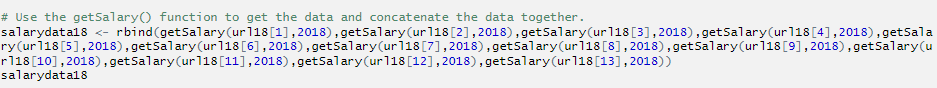
* 1. Change the column names so that it’s easy to join with the stats data later: RK, player, team, salary.
  2. Remove rows such like: 
  3. Under the Name column, separate the name with its position by the comma sign ( , ), and add a new column called position.
  4. Change the salary from character into numeric.
  5. Add a new column called season to indicate the season.

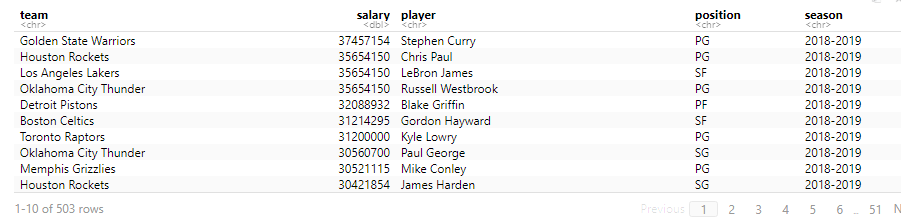
The code of **getSalary()** is shown below:



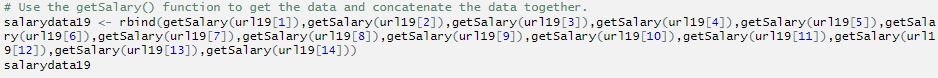
Next, use the function **getSalary()** to scrap the data from ESPN and concatenate the data of different page together.

18/19 season has 503 rows:





19/20 season has 525 rows:





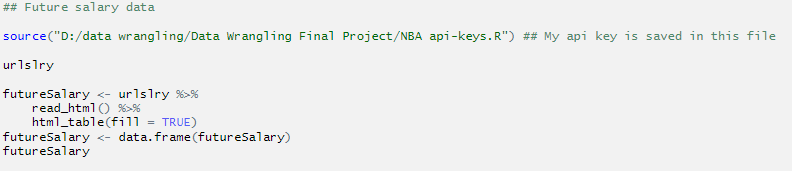
The collecting of salary data of 2018/19 and 2019/2020 seasons is done.

Finally, concatenate the salary data of 2018/19 and 2019/2020 season together with the salary data between the 2002/03 and 2017/18 seasons. The number of rows is correct, there are total of 7283 salary data:

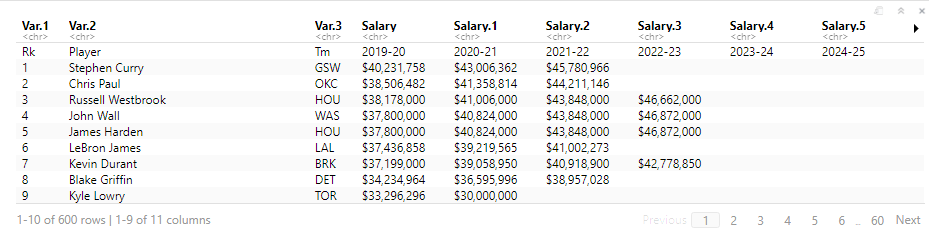


I save the final salary data into csv file: 

In addition, for the future salary data, I use the following code to get it from **Basketball-Reference**. The codeis shown below:

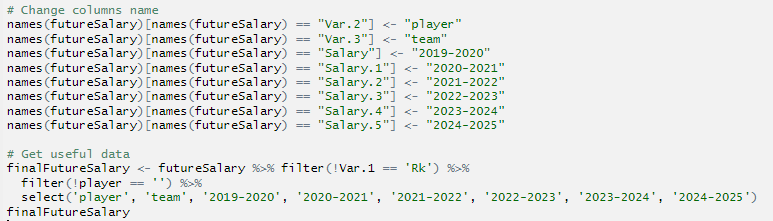


Here is the table:

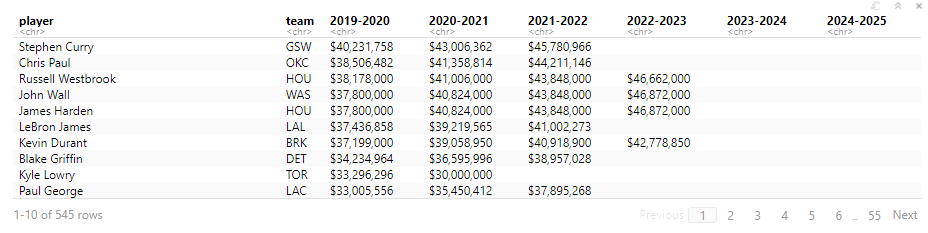


Then, I made several changes:

1. Change the columns name.
2. Remove the useless row by using filter() and select()

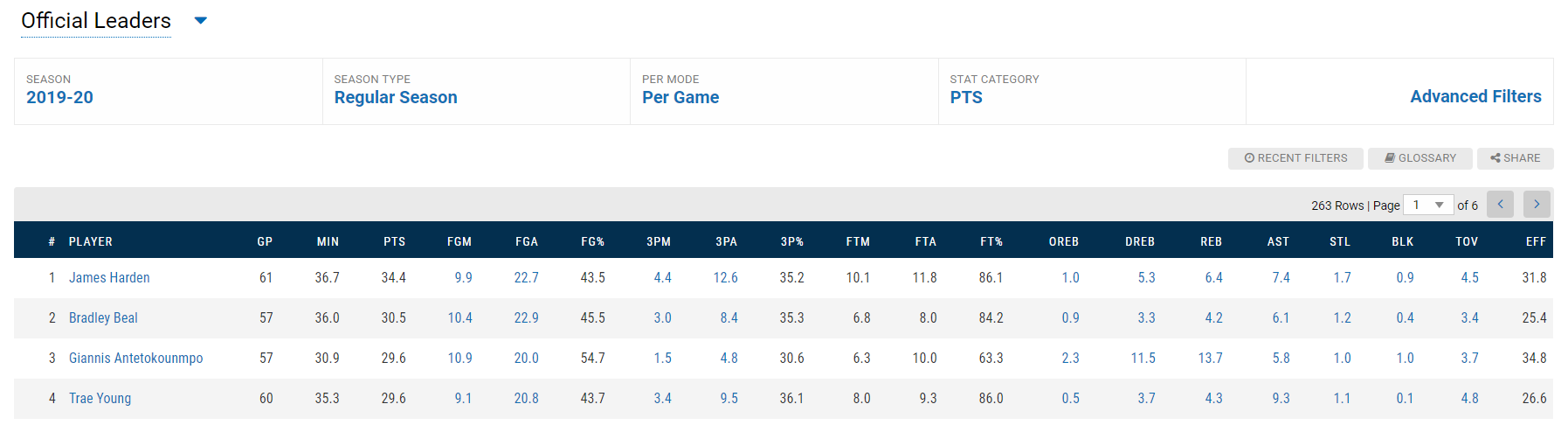
’

The final table of future salary is saved in , and it is shown below:

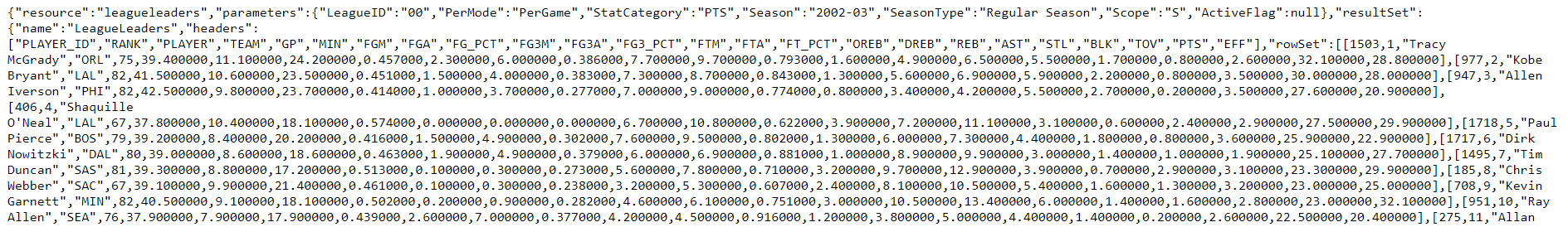


1. *Stats Data*

The stats data is scraped from **NBA official site**:



For every season, I choose to get the data of Regular Season, Per Game, and sort the data by PTS. Unlike the salary data, NBA official site stores the data in JSON format:



The API-keys are written in this form:

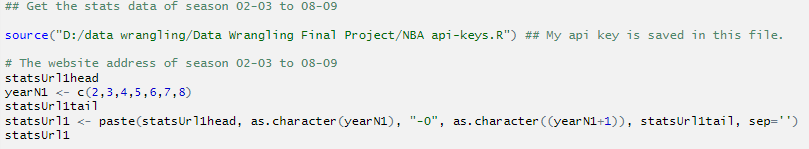
https://stats.nba.com/stats/leagueLeaders?LeagueID=00&PerMode=PerGame&Scope=S&Season=2002-03&SeasonType=Regular+Season&StatCategory=PTS

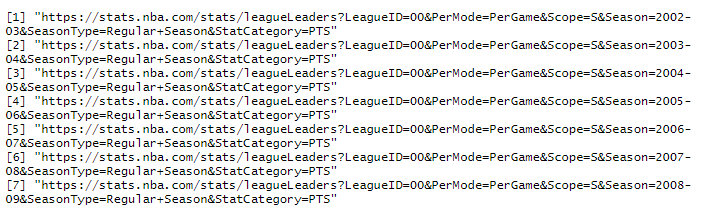
Simply changing the year number after “Season=” will give you the stats data for different season. By finding the pattern of the API-keys, I divide the seasons into 3 parts:

1. 02/03 to 08/09 seasons.
2. 09/10 season.
3. 10/11 to 19/20 seasons.

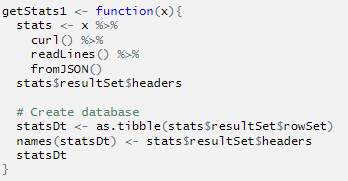
Next, using the same technique, save the API-keys into , implement a code using a vector for different seasons and concatenate them with the common part.

For 02/03 to 08/09 season:

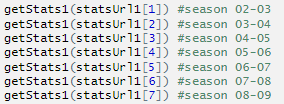




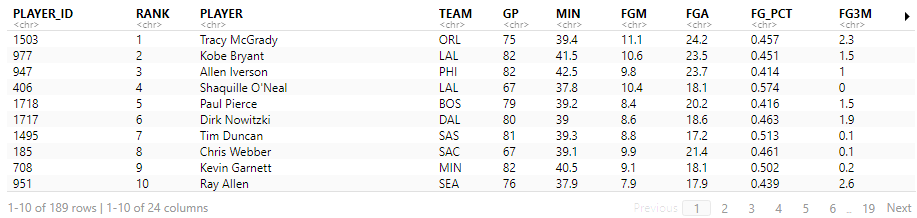
Then write a function **getStats1()** to access and read the JSON file:



Read those files:

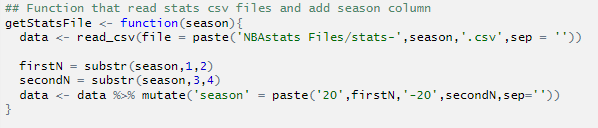


The part of the data of season 02-03 is shown below:

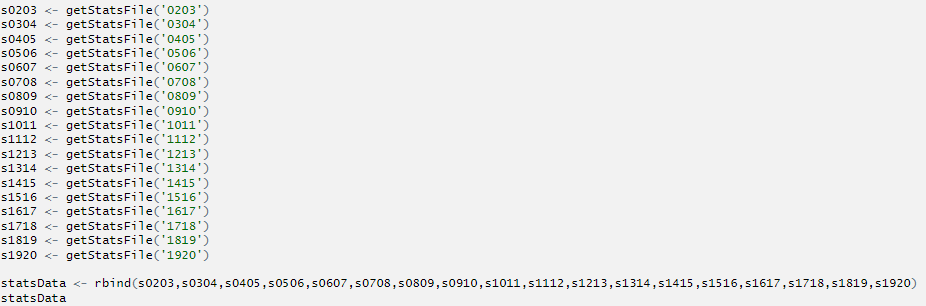


For the rest of the seasons, the approaches are basically the same, so I will not show the codes and images. I have saved the stats data of every season into csv file, under the folder.

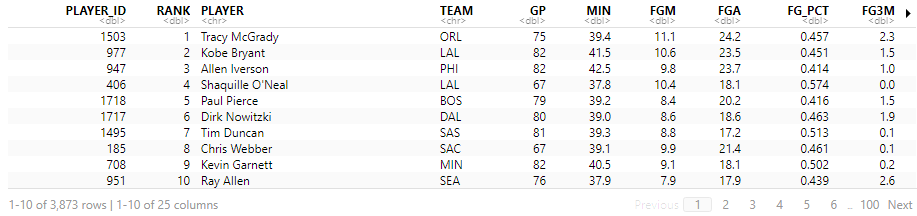
Next, I want to concatenate all the files in together to become a whole data table from 02/03 to 19/20 season. Write a function **getStatsFile()** to read the saved stats file from the folder, in the function we also need to add a new season column as we did with salary data:

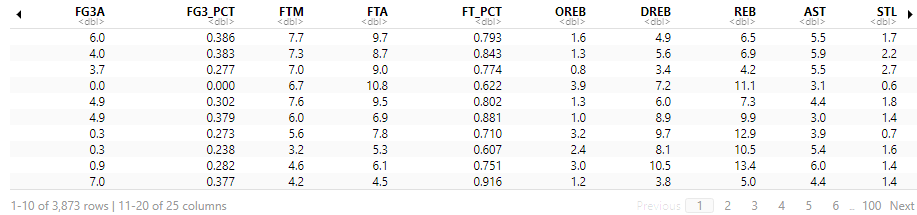


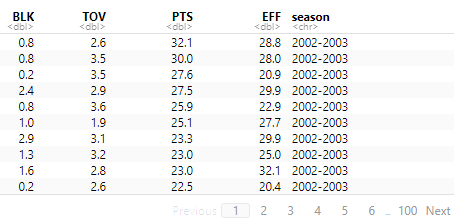
Finally, read and concatenate the stats data, this is the final stats data:



I save the final stats data into . The table is shown below, there are total of 3873 stats data:

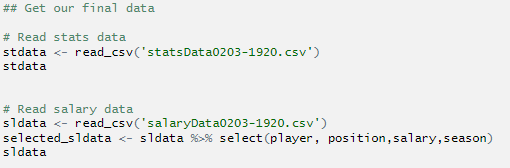




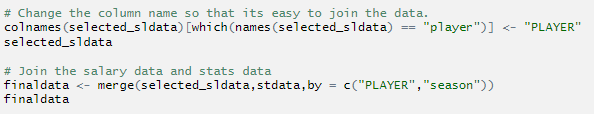


The number of stats data is smaller than the number of salary data is because that there is a Statistical Minimums to qualify for NBA League Leader, which mean a player must play a certain amount of games to get counted in stats data. So, not all the players are recorded in the stats data.

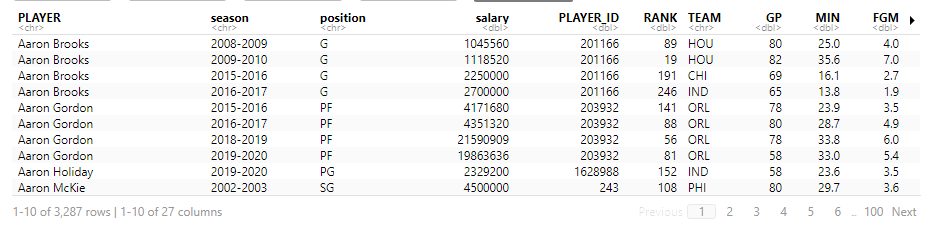
Last step is to get our final whole data. First, read the final **salary data** and the final **stats data** from the csv files:

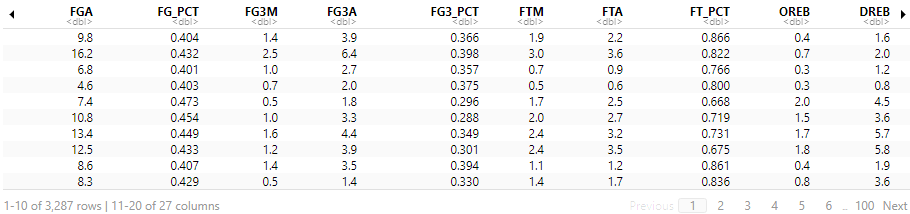


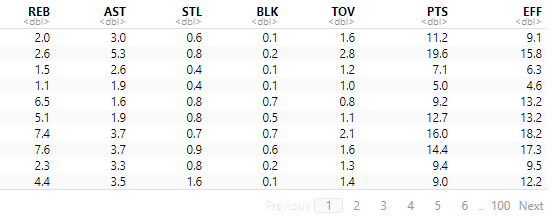
Second, integrate the column names so that they are easy to join. Notice that we need to join two data by player name and season:



After joining, we successfully get the data we want:







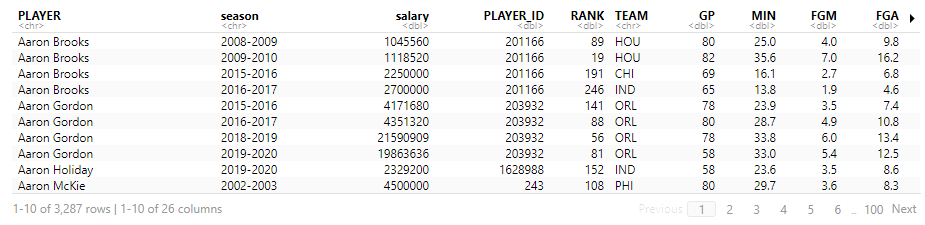
Save this table into , this is our final data!

**Data Analysis:**

Now it is time to investigate the data using R code. The code for data analyzing is saved in  . This is not the main focus of our project, so I will display shortly. First, load some packages that are needed for this project:

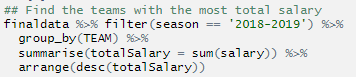


Second, read the file:



With these data, we can do what we want.

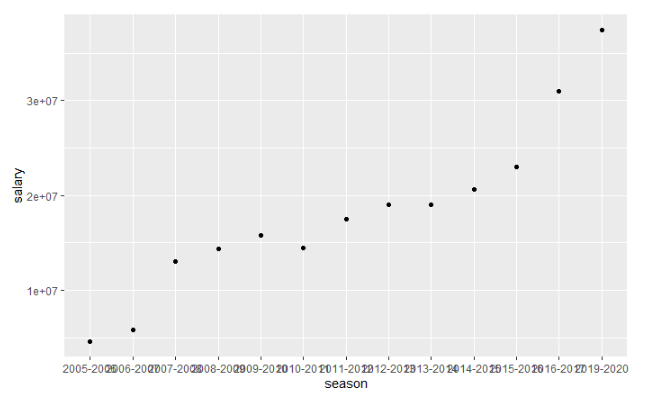
1. Find the teams with the most total salary in 18/19 season:





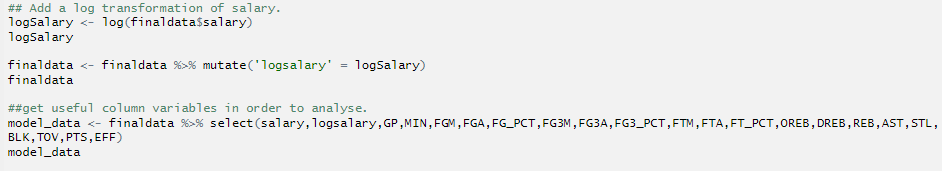
1. Get the salary graph of LeBron James:





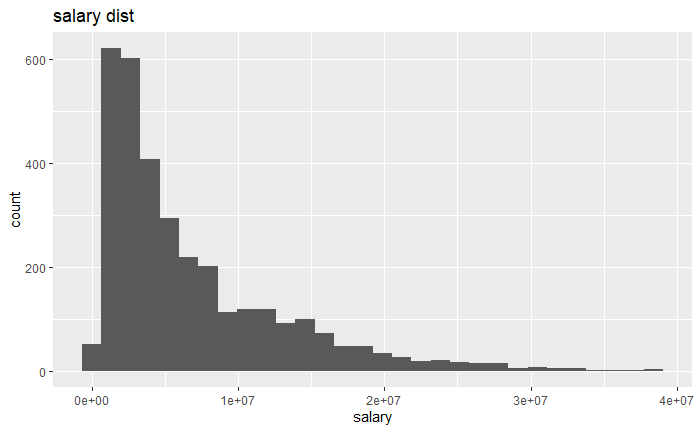
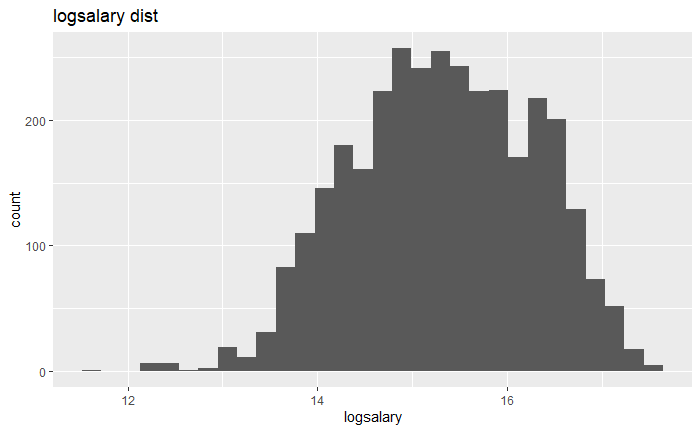
1. Get the graphs of distribution of different stats:

First, apply the log transformation to the salary data, and select the stats for us to analyze:



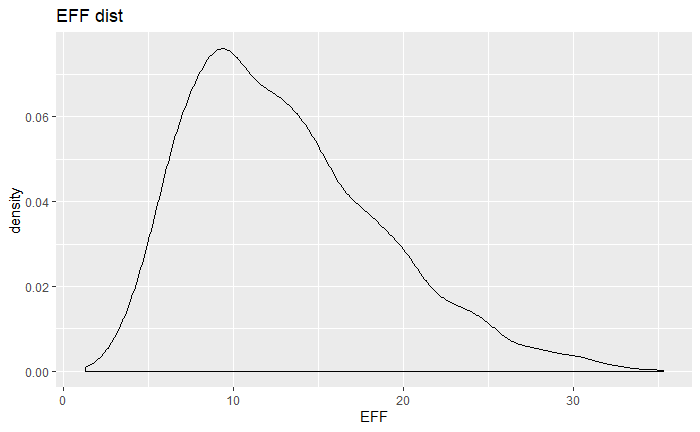
Second, try histogram on salary data and log transformation of salary data:

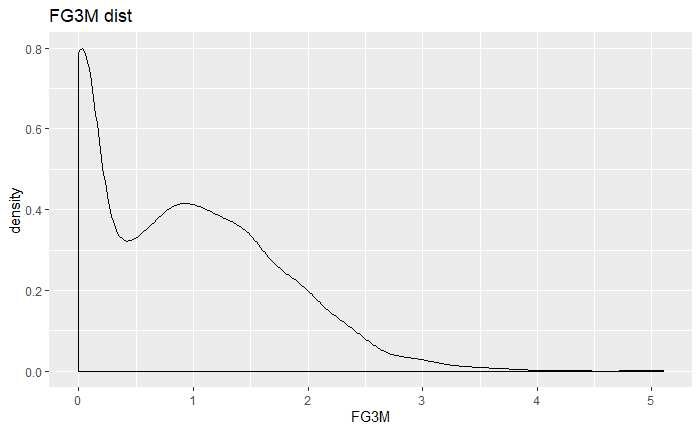
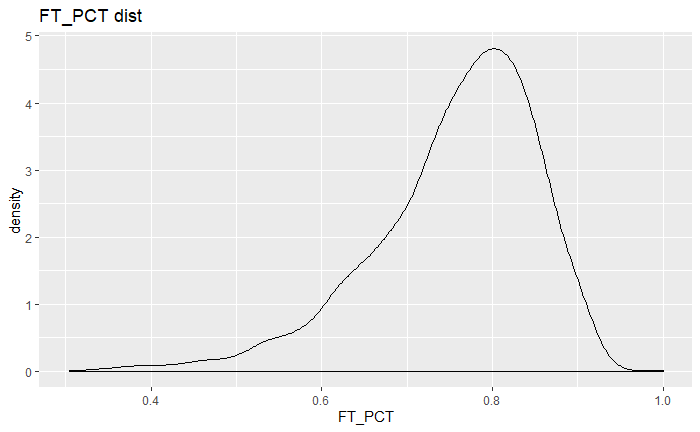


Third, try density plots of different stats:



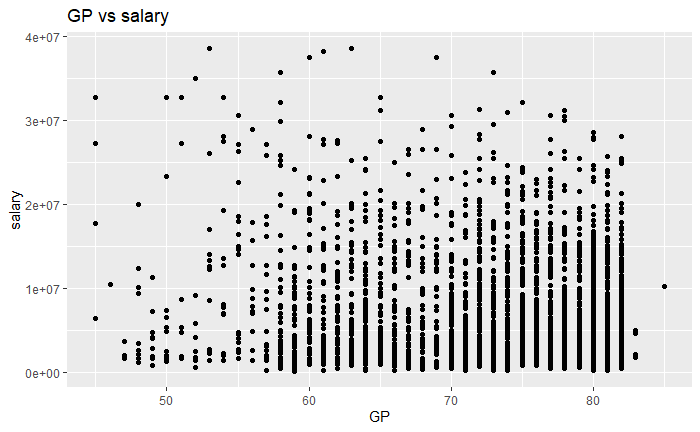
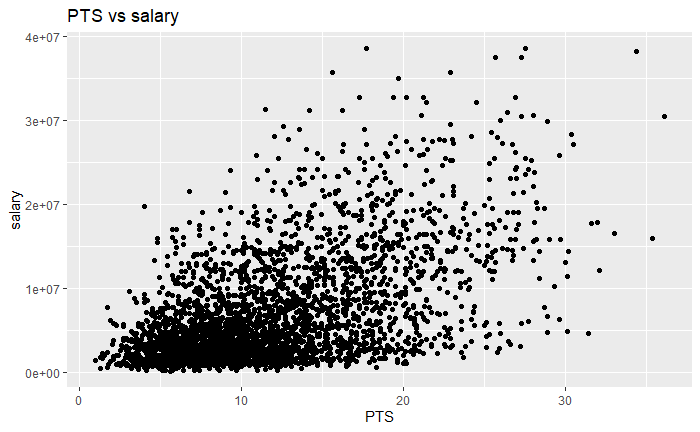
 

Next, investigate the relationship between salary and other stats:



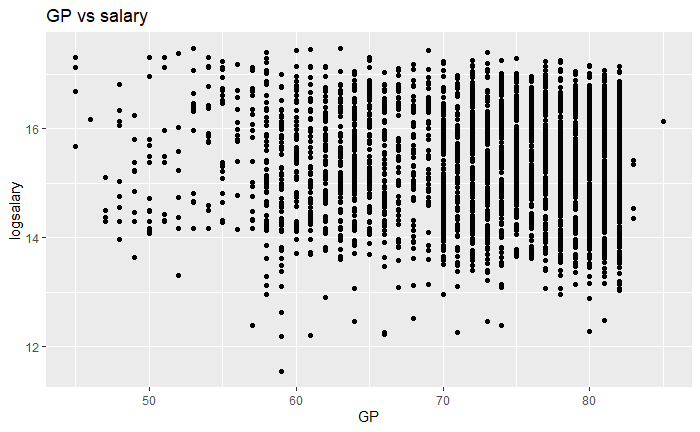
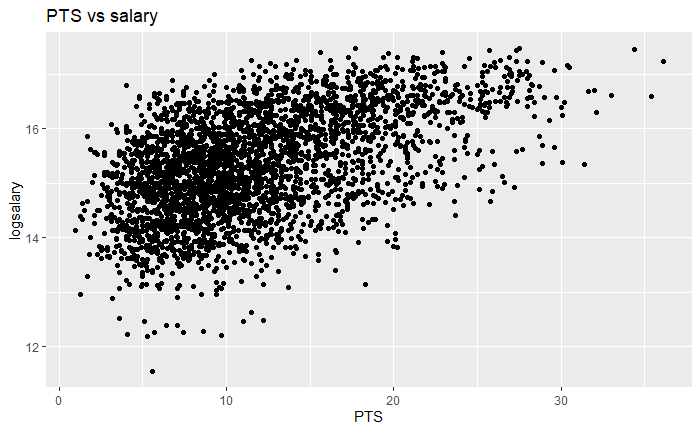


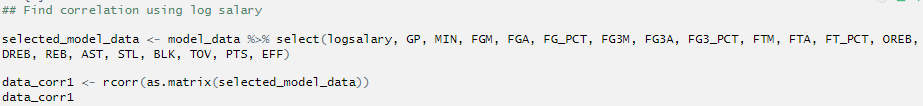
Try log salary:

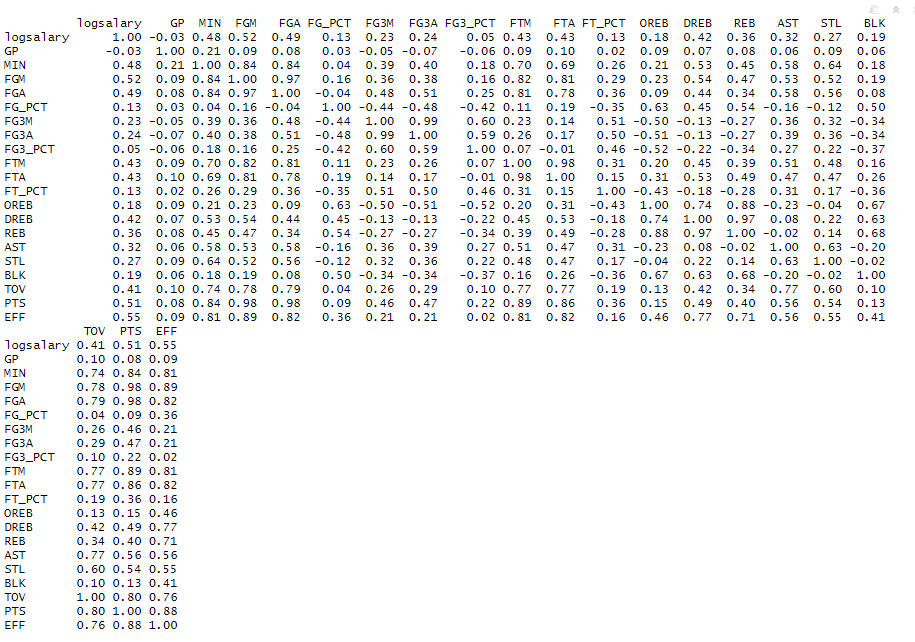




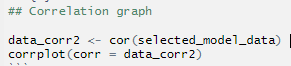
 

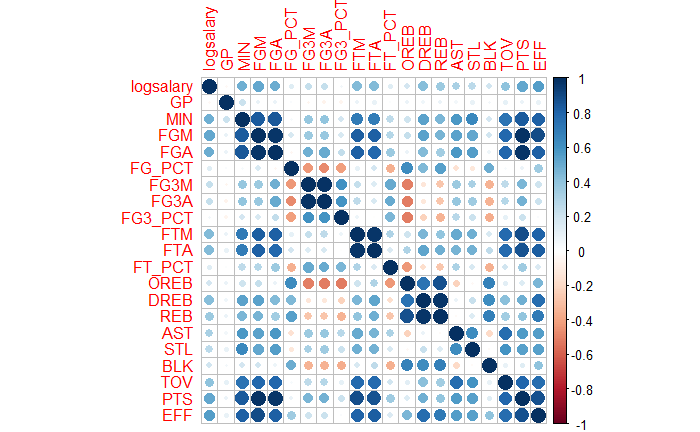
1. Find some correlation between salary and stats:





Make correlation graph:





**Conclusion:**

In order to finish this project, I use a lot of knowledges that learned from the class, including:

1. Loading different packages for different uses.
2. Filtering, selecting, grouping, arranging, concatenating, joining, and summarizing the data.
3. Import and export different files such as csv file and JSON file.
4. Write some function to reduce the repeated work.
5. Manipulate strings such as URL address and API-keys.
6. Scraping tables and data from the website using URL address and API-keys.
7. Interact with different R files.
8. Make graphs by using ggplot.
9. Project in R.
10. Github.

In conclusion, by doing this project, I have not only reviewed my class notes, practiced my skill, but also learned a lot of new stuff. It is a really good experience for me.