NBA’s 2021-2022 Spalding to Wilson Ball Transition
R Analysis

Ricky Camilo
June 2023

Table of Contents

Useful Basketball Statistics Terminology:

GM, GP, GS- Games, Games Played, Games Started

PTS- Points

FGM, FGA, pctFG- Field Goals Made, Attempted and Percentage

FTM, FTA, pctFT- Free Throws Made, Attempted and Percentage

3FGM, 3FGA, pctFG3- Three-Point Field Goals Made, Attempted and Percentage

REB, OREB, DREB- Rebounds, Offensive Rebounds, Defensive Rebounds

AST- Assists

STL- Steals

BLK- Blocks

TO- Turnovers

TD- Triple Double

PF- Personal Fouls

MIN- Minutes

AST/TO- Assist to Turnover Ratio

PER- Player Efficiency Rating

ORtg- Advanced Offensive Rating

DRtg- Advanced Defensive Rating

NRtg- Advanced Net Rating

What’s so Interesting About this Switch?

The transition from Spalding basketballs to Wilson basketballs in the NBA marked a significant change for the league and its players. For over three decades, Spalding had been the official basketball supplier for the NBA, but in 2021, the league announced a new partnership with Wilson Sporting Goods. This switch brought/bring(s) about several noteworthy adjustments in the game. "The NBA players are incredibly skilled, so even very small, subtle changes can take time to adjust to. And, in some cases, maybe impossible to adjust their playing styles to, he said. Since it's still a leather ball, my guess would be that the players would adapt and adjust to it after a while. Now, how long is 'after a while? Is it weeks? Is it months? That I don't know. There is no word yet on the future of this ball. But for now, it seems like the league will wait and adjust.” Sundaresan. (2001).

Players noticed a slightly different texture and bounce, requiring some time to adjust their shooting and dribbling techniques accordingly. This shift to Wilson equipment also prompted a period of adaptation for teams as they incorporated the new basketballs into their training and game strategies. Additionally, the transition sparked discussions about the impact on shooting percentages and overall game performance, with players expressing mixed opinions on the change. “Not to make an excuse about the ball, but it’s a different basketball. It doesn’t have the same touch/softness that the Spalding ball had, and you’ll see this year. It’s gonna be a lot of bad misses,” said Clippers forwards Paul George. Chow. (2001).

Wilson who intended to make the new ball as close as possible to the Spalding one has had a lot to answer to since it took over. “The NBA and its players will work jointly with Wilson to develop and approve the new game ball.” Wilson says the ball is almost identical to the Spalding ball as they didn’t try to change anything. “Wilson will manufacture the NBA, WNBA and NBA G League game balls using the same materials, eight-panel configuration and performance specifications as current game balls and will also source the same leather currently used in the NBA,” said the NBA. Mangubat. (2020).

This partnership emphasized the NBA's pursuit of enhancing the game's integrity and maintaining the highest standards of play. With the introduction of Wilson basketballs, the NBA entered a new era, marked by a familiar but fresh collaboration but also criticism from its own players. Does the statistical data agree or disagree with the players grievances?

https://github.com/rickycamilo/-R-Analysis-Spalding-to-Wilson-Ball-Transition

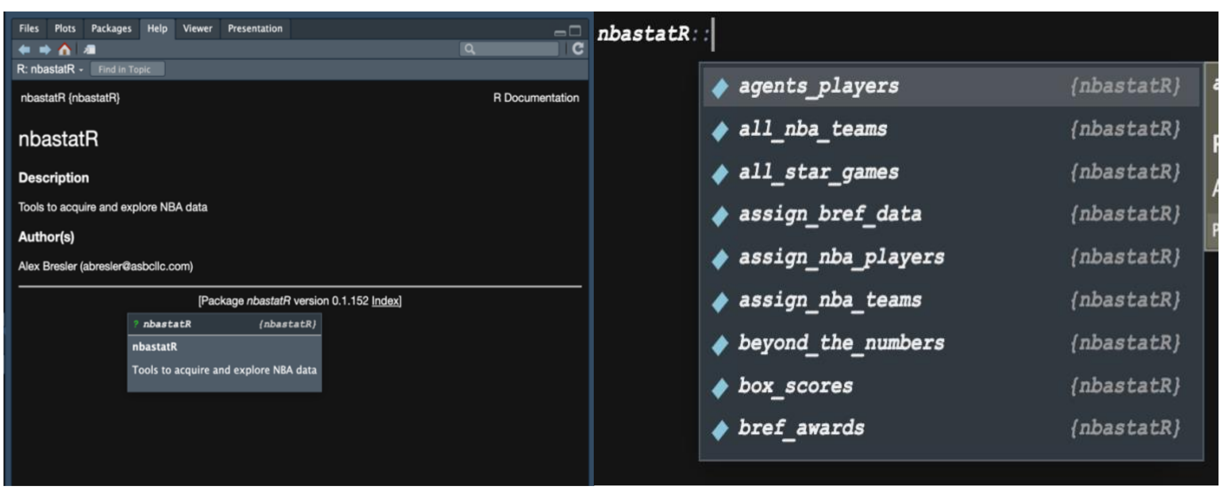
How Will We Analyze and Munge This Data?

NBAStatR is an R library specifically designed for analyzing and extracting insights from NBA basketball data. It serves as a valuable resource for statisticians, analysts, and enthusiasts interested in diving deep into the world of basketball analytics. **NBAStatR** provides a wide range of functions and tools to efficiently access, preprocess, and analyze NBA data, making it an indispensable tool for understanding the intricacies of the game.

One of the notable features of **NBAStatR** is its ability to retrieve and import NBA data directly from various sources, including popular APIs and databases. This streamlined data access saves analysts time and effort, allowing them to focus on the analytical aspects of their work. In our case we will be using the default basketballreference.com which is the golden standard for comprehensive NBA data.

We will be using **NBAStatR** in conjunction with **dplyr** and **ggplot2** in R Studio to clean, filter, visualize and do some statistical measurements to really get to know our data. **Dplyr** is a powerful R library that provides a wide range of functions for efficient data manipulation and cleaning. When it comes to data cleaning, **dplyr** offers several essential functions that facilitate the process and ensure data integrity. With **ggplot2**, you can create a variety of plots, including scatter plots, line plots, bar charts, histograms, box plots, and more. It provides a flexible and intuitive syntax that allows users to specify the data, aesthetics (such as mapping variables to visual attributes like color or size), and layers of a plot.

We will later analyze some advanced analytics using a web scraper and put our clean and shaped data into a K-means clustering algorithm to see if we recognize any patterns that might support the players claims that the new ball is responsible for the slight scoring slump early in the 2021-2022 season.



Analyzing Per Game Scoring Data From Spaulding Ball Seasons vs. Wilson

First let’s begin by calling in our libraries that we will use in this analysis. Then we set the variable environment to the power of 2 as this specific Mac version of R Studio 2023.03.1+446 environment variable size is not large enough by default.

```
library(dplyr)
library(ggplot2)
library(nbastatR)
```

```
Sys.setenv("VROOM_CONNECTION_SIZE" = 131072 * 2)
```

We now start by creating our 4 data frames for each of the 2019-2020, 2020-2021, 2021-2022, 2022-2023. The **NBAstatR** library has a range of different metrics you can import but for this analysis we will use **metrics_leaders.** Set the seasons to each respective second half of each season’s year. Set the metric equal to “pts”, season_types = “Regular Season”, modes to = c(“PerGame”), return_message to = T, and nest_data to = F

```
# dataframe with stats for each of the last 4 seasons

season2019_20 <- nbastatR::metrics_leaders(seasons = 2020, metric = "pts", season_types
= "Regular Season", modes = c("PerGame"), return_message = TRUE, nest_data =
FALSE)

season2020_21 <- nbastatR::metrics_leaders(seasons = 2021, metric = "pts", season_types
= "Regular Season", modes = c("PerGame"), return_message = TRUE, nest_data =
FALSE)

season2021_22 <- nbastatR::metrics_leaders(seasons = 2022, metric = "pts", season_types
= "Regular Season", modes = c("PerGame"), return_message = TRUE, nest_data =
FALSE)

season2022_23 <- nbastatR::metrics_leaders(seasons = 2023, metric = "pts", season_types
= "Regular Season", modes = c("PerGame"), return_message = TRUE, nest_data =
FALSE
```

We then take those data frames and filter each for players who played at least 58 games in a season. Reason for this is to exclude any players that are extreme outliers (i.e.: players who may have checked in for >2 minutes the entire season and only took one FT therefore is shooting 100%) 58 games is also the cut off point for qualifying for any annual NBA awards.

```
# data frames filtered for players who played 58 games and qualify as a full season and
qualify for awards
```

```
season2019_20 <- filter(season2019_20, gp >=58)
season2020_21 <- filter(season2020_21, gp >=58)
season2021_22 <- filter(season2021_22, gp >=58)
season2022_23 <- filter(season2022_23, gp >=58)
```

2019-2020:

TATR.R* | season2019_20 | season2020_21 | season2021_22 | season2022_23 | P_gamelog_all

Filter

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam
1	PerGame	pts	2019–20	Regular Season	S	James Harden	HOU
2	PerGame	pts	2019–20	Regular Season	S	Damian Lillard	POR
3	PerGame	pts	2019–20	Regular Season	S	Trae Young	ATL
4	PerGame	pts	2019–20	Regular Season	S	Giannis Antetokounmpo	MIL
5	PerGame	pts	2019–20	Regular Season	S	Luka Doncic	DAL
6	PerGame	pts	2019–20	Regular Season	S	Devin Booker	PHX
7	PerGame	pts	2019–20	Regular Season	S	Anthony Davis	LAL
8	PerGame	pts	2019–20	Regular Season	S	Zach LaVine	CHI
9	PerGame	pts	2019–20	Regular Season	S	LeBron James	LAL
10	PerGame	pts	2019–20	Regular Season	S	Donovan Mitchell	UTA
11	PerGame	pts	2019–20	Regular Season	S	Brandon Ingram	NOP
12	PerGame	pts	2019–20	Regular Season	S	Jayson Tatum	BOS
13	PerGame	pts	2019–20	Regular Season	S	Pascal Siakam	TOR
14	PerGame	pts	2019–20	Regular Season	S	CJ McCollum	POR
15	PerGame	pts	2019–20	Regular Season	S	DeMar DeRozan	SAS
16	PerGame	pts	2019–20	Regular Season	S	Khris Middleton	MIL
17	PerGame	pts	2019–20	Regular Season	S	Collin Sexton	CLE
18	PerGame	pts	2019–20	Regular Season	S	Spencer Dinwiddie	BKN
19	PerGame	pts	2019–20	Regular Season	S	Bojan Bogdanovic	UTA
20	PerGame	pts	2019–20	Regular Season	S	Jimmy Butler	MIA
21	PerGame	pts	2019–20	Regular Season	S	Nikola Jokic	DEN

2020-2021:

TATR.R* | season2019_20 | season2020_21 | season2021_22 | season2022_23 | P_gamelog_all | ml p

Filter

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam	idPl
1	PerGame	pts	2020–21	Regular Season	S	Stephen Curry	GSW	
2	PerGame	pts	2020–21	Regular Season	S	Bradley Beal	WAS	
3	PerGame	pts	2020–21	Regular Season	S	Damian Lillard	POR	
4	PerGame	pts	2020–21	Regular Season	S	Giannis Antetokounmpo	MIL	
5	PerGame	pts	2020–21	Regular Season	S	Luka Doncic	DAL	1
6	PerGame	pts	2020–21	Regular Season	S	Zach LaVine	CHI	
7	PerGame	pts	2020–21	Regular Season	S	Zion Williamson	NOP	1
8	PerGame	pts	2020–21	Regular Season	S	Jayson Tatum	BOS	1
9	PerGame	pts	2020–21	Regular Season	S	Nikola Jokic	DEN	
10	PerGame	pts	2020–21	Regular Season	S	Devin Booker	PHX	1
11	PerGame	pts	2020–21	Regular Season	S	Trae Young	ATL	1
12	PerGame	pts	2020–21	Regular Season	S	De'Aaron Fox	SAC	1
13	PerGame	pts	2020–21	Regular Season	S	Jaylen Brown	BOS	1
14	PerGame	pts	2020–21	Regular Season	S	Collin Sexton	CLE	1
15	PerGame	pts	2020–21	Regular Season	S	Julius Randle	NYK	
16	PerGame	pts	2020–21	Regular Season	S	Brandon Ingram	NOP	1
17	PerGame	pts	2020–21	Regular Season	S	Nikola Vucevic	CHI	
18	PerGame	pts	2020–21	Regular Season	S	Russell Westbrook	WAS	
19	PerGame	pts	2020–21	Regular Season	S	DeMar DeRozan	SAS	
20	PerGame	pts	2020–21	Regular Season	S	Terry Rozier	CHA	1
21	PerGame	pts	2020–21	Regular Season	S	Khris Middleton	MIL	

Showing 1 to 21 of 171 entries, 33 total columns

2021-2022

ATR.R* × | season2019_20 × | season2020_21 × | season2021_22 × | season2022_23 × | P_gamelog_all ×

Filter

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam
1	PerGame	pts	2021–22	Regular Season	S	Joel Embiid	PHI
2	PerGame	pts	2021–22	Regular Season	S	Giannis Antetokounmpo	MIL
3	PerGame	pts	2021–22	Regular Season	S	Luka Doncic	DAL
4	PerGame	pts	2021–22	Regular Season	S	Trae Young	ATL
5	PerGame	pts	2021–22	Regular Season	S	DeMar DeRozan	CHI
6	PerGame	pts	2021–22	Regular Season	S	Nikola Jokic	DEN
7	PerGame	pts	2021–22	Regular Season	S	Jayson Tatum	BOS
8	PerGame	pts	2021–22	Regular Season	S	Devin Booker	PHX
9	PerGame	pts	2021–22	Regular Season	S	Donovan Mitchell	UTA
10	PerGame	pts	2021–22	Regular Season	S	Stephen Curry	GSW
11	PerGame	pts	2021–22	Regular Season	S	Karl–Anthony Towns	MIN
12	PerGame	pts	2021–22	Regular Season	S	Zach LaVine	CHI
13	PerGame	pts	2021–22	Regular Season	S	Jaylen Brown	BOS
14	PerGame	pts	2021–22	Regular Season	S	De'Aaron Fox	SAC
15	PerGame	pts	2021–22	Regular Season	S	Pascal Siakam	TOR
16	PerGame	pts	2021–22	Regular Season	S	CJ McCollum	NOP
17	PerGame	pts	2021–22	Regular Season	S	James Harden	PHI
18	PerGame	pts	2021–22	Regular Season	S	Darius Garland	CLE
19	PerGame	pts	2021–22	Regular Season	S	Anthony Edwards	MIN
20	PerGame	pts	2021–22	Regular Season	S	Dejounte Murray	SAS
21	PerGame	pts	2021–22	Regular Season	S	Tyler Herro	MIA

Showing 1 to 21 of 228 entries, 33 total columns

2022-2023

TATR.R* × | season2019_20 × | season2020_21 × | season2021_22 × | season2022_23 × | P_gamelog_all ×

Filter

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam	id
1	PerGame	pts	2022–23	Regular Season	S	Joel Embiid	PHI	
2	PerGame	pts	2022–23	Regular Season	S	Luka Doncic	DAL	
3	PerGame	pts	2022–23	Regular Season	S	Damian Lillard	POR	
4	PerGame	pts	2022–23	Regular Season	S	Shai Gilgeous–Alexander	OKC	
5	PerGame	pts	2022–23	Regular Season	S	Giannis Antetokounmpo	MIL	
6	PerGame	pts	2022–23	Regular Season	S	Jayson Tatum	BOS	
7	PerGame	pts	2022–23	Regular Season	S	Donovan Mitchell	CLE	
8	PerGame	pts	2022–23	Regular Season	S	Kyrie Irving	DAL	
9	PerGame	pts	2022–23	Regular Season	S	Jaylen Brown	BOS	
10	PerGame	pts	2022–23	Regular Season	S	Trae Young	ATL	
11	PerGame	pts	2022–23	Regular Season	S	Ja Morant	MEM	
12	PerGame	pts	2022–23	Regular Season	S	Lauri Markkanen	UTA	
13	PerGame	pts	2022–23	Regular Season	S	Julius Randle	NYK	
14	PerGame	pts	2022–23	Regular Season	S	De'Aaron Fox	SAC	
15	PerGame	pts	2022–23	Regular Season	S	Zach LaVine	CHI	
16	PerGame	pts	2022–23	Regular Season	S	Anthony Edwards	MIN	
17	PerGame	pts	2022–23	Regular Season	S	DeMar DeRozan	CHI	
18	PerGame	pts	2022–23	Regular Season	S	Nikola Jokic	DEN	
19	PerGame	pts	2022–23	Regular Season	S	Pascal Siakam	TOR	
20	PerGame	pts	2022–23	Regular Season	S	Jalen Brunson	NYK	
21	PerGame	pts	2022–23	Regular Season	S	Kristaps Porzingis	WAS	

Showing 1 to 21 of 245 entries, 33 total columns

Since we are looking for any noticeable differences in shooting, we're going to focus on 4 metrics: pctFG, pctFT, pctFG3, PTS (Field Goal %, Free Throw %, 3 Point Field Goal %, Points).

We look for the arithmetic mean of each column for the above-mentioned fields for each season. As we can see the average field goal percentage for the 2019-2020 season was 46.7%, then increased to 47.8% in the final Spalding season then slightly dipped to 46.9% in the first Wilson ball season then up to 48.3% in the 2022-2023 season. These are not significantly different from each other so it's hard to make a judgment based on Field Goal %.

In the 2019-2020 season the average free throw percentage was 77.4% then 77.5% in the final Spalding season, then 77.5% in the first Wilson season and finally 77.3% in the 2022-2023 season. Nothing that jumps analyzing average free throw percentage.

Shooting from 3 range was mostly the same too starting with 33.9% in the 2019-2020 season then 33.2%, then 33.1%, and 32.7% in the 2022-2023 season. Next up is points where 12.2 was the average points scored in the NBA by player then 12.3, then 12.2 and finally 12.6 in the 2022-2023 season which is almost a half a point increase but still nothing noticeable.

```
# average fg%, ft%, and ppg, and 3pm%

mean(season2019_20$pctFG)
[1] 0.4666243
mean(season2020_21$pctFG)
[1] 0.4785965
mean(season2021_22$pctFG)
[1] 0.4692632
mean(season2022_23$pctFG)
[1] 0.4835959


mean (season2019_20$pctFT)
[1] 0.7740794
mean(season2020_21$pctFT)
[1] 0.7753041
mean(season2021_22$pctFT)
[1] 0.7753158
mean(season2022_23$pctFT)
[1] 0.7739837


mean(season2019_20$pctFG3)
[1] 0.3388266
mean(season2020_21$pctFG3)
[1] 0.3325816
```

```
mean(season2021_22$pctFG3)
[1] 0.3319681
mean(season2022_23$pctFG3)
[1] 0.3271997
```

```
mean(season2019_20$pts)
[1] 12.18519
mean(season2020_21$pts)
[1] 12.29708
mean(season2021_22$pts)
[1] 12.1886
mean(season2022_23$pts)
[1] 12.62735
```

The Hype Beast article: NBA Players Comment on New Wilson Game Basketball mentioned a 2% decrease in field goal percentage at the beginning of the 2021-2022 season and it looks like post that article publishing the NBA collective shot about 1% better the rest of the season. So far, the ball switch has not made a quantitative difference in player statistic; if anything, it seems like scoring and efficiency is trending upwards.

It will become easier to work and analyze this NBA data if we combine these 4 seasons into 2 data frames. One for each ball manufacturer. We can use the **rbind** function for that. The first data frame will be a combination of the 2019-2020 and 2020-2021 seasons (Spalding Data) and then the other data frame will be the 2021-2022 and 2022-2023 seasons (Wilson Data).

```
# combined dataframes for spalding 19-21 and wilson 21-23

spalding_data <- rbind(season2019_20,season2020_21)
wilson_data <- rbind(season2021_22,season2022_23)
```

Spalding:

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam	idPlayer	numberRank	idTeam	gp	fgm
1	PerGame	pts	2020-21	Regular Season	S	Stephen Curry	GSW	201939	1	1610612744	63	10
2	PerGame	pts	2019-20	Regular Season	S	James Harden	HOU	201935	1	1610612745	68	9
3	PerGame	pts	2020-21	Regular Season	S	Bradley Beal	WAS	203078	2	1610612764	60	11
4	PerGame	pts	2020-21	Regular Season	S	Damian Lillard	POR	203081	3	1610612757	67	9
5	PerGame	pts	2019-20	Regular Season	S	Damian Lillard	POR	203081	3	1610612757	66	9
6	PerGame	pts	2019-20	Regular Season	S	Trae Young	ATL	1629027	4	1610612737	60	9
7	PerGame	pts	2020-21	Regular Season	S	Giannis Antetokounmpo	MIL	203507	5	1610612749	61	10
8	PerGame	pts	2019-20	Regular Season	S	Giannis Antetokounmpo	MIL	203507	5	1610612749	63	10
9	PerGame	pts	2020-21	Regular Season	S	Luka Doncic	DAL	1629029	6	1610612742	66	9
10	PerGame	pts	2019-20	Regular Season	S	Luka Doncic	DAL	1629029	6	1610612742	61	9
11	PerGame	pts	2020-21	Regular Season	S	Zach LaVine	CHI	203897	7	1610612741	58	9
12	PerGame	pts	2020-21	Regular Season	S	Zion Williamson	NOP	1629627	8	1610612740	61	10
13	PerGame	pts	2019-20	Regular Season	S	Devin Booker	PHX	1626164	9	1610612756	70	9
14	PerGame	pts	2020-21	Regular Season	S	Jayson Tatum	BOS	1628369	10	1610612738	64	9
15	PerGame	pts	2019-20	Regular Season	S	Anthony Davis	LAL	203076	10	1610612747	62	8
16	PerGame	pts	2019-20	Regular Season	S	Zach LaVine	CHI	203897	11	1610612741	60	9
17	PerGame	pts	2020-21	Regular Season	S	Nikola Jokic	DEN	203999	12	1610612743	72	10
18	PerGame	pts	2019-20	Regular Season	S	LeBron James	LAL	2544	12	1610612747	67	9
19	PerGame	pts	2020-21	Regular Season	S	Devin Booker	PHX	1626164	13	1610612756	67	9
20	PerGame	pts	2019-20	Regular Season	S	Donovan Mitchell	UTA	1628378	13	1610612762	69	8
21	PerGame	pts	2020-21	Regular Season	S	Trae Young	ATL	1629027	14	1610612737	63	7

Showing 1 to 21 of 360 entries, 33 total columns

Wilson:

	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam	idPlayer	numberRank	idTeam	gp
1	PerGame	pts	2021-22	Regular Season	S	Joel Embiid	PHI	203954	1	1610612755	68
2	PerGame	pts	2022-23	Regular Season	S	Joel Embiid	PHI	203954	1	1610612755	66
3	PerGame	pts	2021-22	Regular Season	S	Giannis Antetokounmpo	MIL	203507	2	1610612749	67
4	PerGame	pts	2022-23	Regular Season	S	Luka Doncic	DAL	1629029	2	1610612742	66
5	PerGame	pts	2021-22	Regular Season	S	Luka Doncic	DAL	1629029	3	1610612742	65
6	PerGame	pts	2022-23	Regular Season	S	Damian Lillard	POR	203081	3	1610612757	58
7	PerGame	pts	2021-22	Regular Season	S	Trae Young	ATL	1629027	4	1610612737	76
8	PerGame	pts	2022-23	Regular Season	S	Shai Gilgeous-Alexander	OKC	1628983	4	1610612760	68
9	PerGame	pts	2021-22	Regular Season	S	DeMar DeRozan	CHI	201942	5	1610612741	76
10	PerGame	pts	2022-23	Regular Season	S	Giannis Antetokounmpo	MIL	203507	5	1610612749	63
11	PerGame	pts	2021-22	Regular Season	S	Nikola Jokic	DEN	203999	6	1610612743	74
12	PerGame	pts	2022-23	Regular Season	S	Jayson Tatum	BOS	1628369	6	1610612738	74
13	PerGame	pts	2021-22	Regular Season	S	Jayson Tatum	BOS	1628369	7	1610612738	76
14	PerGame	pts	2022-23	Regular Season	S	Donovan Mitchell	CLE	1628378	7	1610612739	68
15	PerGame	pts	2021-22	Regular Season	S	Devin Booker	PHX	1626164	8	1610612756	68
16	PerGame	pts	2022-23	Regular Season	S	Kyrie Irving	DAL	202681	8	1610612742	60
17	PerGame	pts	2021-22	Regular Season	S	Donovan Mitchell	UTA	1628378	9	1610612762	67
18	PerGame	pts	2022-23	Regular Season	S	Jaylen Brown	BOS	1627759	9	1610612738	67
19	PerGame	pts	2021-22	Regular Season	S	Stephen Curry	GSW	201939	10	1610612744	64
20	PerGame	pts	2022-23	Regular Season	S	Trae Young	ATL	1629027	10	1610612737	73
21	PerGame	pts	2021-22	Regular Season	S	Karl-Anthony Towns	MIN	1626157	11	1610612750	74

Showing 1 to 21 of 473 entries, 33 total columns

For the sake of the K-means clusters we will build later we will build a data frame that combines all 4 seasons together by combining the **spalding_data** *and* **wilson_data**

last_four_seasons_data <- rbind(spalding_data,wilson_data)

Here we truncate the data frame to have only columns we will need for our analysis and algorithm.

last_four_seasons_truncated <-
select(last_four_seasons_data,pctFG,pts,pctFG3,pctFT,namePlayer,slugSeason)

season2021_22 | season2022_23 | P_gamelog_all | ml project k means clustering.R* | last_four_seasons_truncated

Filter

	pctFG	pts	pctFG3	pctFT	namePlayer	slugSeason
1	0.437	14.4	0.3157895	0.674	Aaron Gordon	2019-20
2	0.520	15.0	0.3428571	0.743	Aaron Gordon	2021-22
3	0.564	16.3	0.3600000	0.608	Aaron Gordon	2022-23
4	0.414	9.5	0.3939394	0.851	Aaron Holiday	2019-20
5	0.390	7.2	0.3571429	0.819	Aaron Holiday	2020-21
6	0.447	6.3	0.3750000	0.868	Aaron Holiday	2021-22
7	0.418	3.9	0.4285714	0.844	Aaron Holiday	2022-23
8	0.427	10.1	0.3720930	0.838	Aaron Nesmith	2022-23
9	0.512	6.8	0.4117647	0.831	Aaron Wiggins	2022-23
10	0.465	8.9	0.3888889	0.894	AJ Griffin	2022-23
11	0.450	11.9	0.3571429	0.763	Al Horford	2019-20
12	0.467	10.2	0.3421053	0.842	Al Horford	2021-22
13	0.476	9.8	0.4423077	0.714	Al Horford	2022-23
14	0.418	15.0	0.3913043	0.887	Alec Burks	2019-20
15	0.391	11.7	0.3958333	0.822	Alec Burks	2021-22
16	0.408	7.6	0.2812500	0.700	Aleksej Pokusevski	2021-22
17	0.412	5.5	0.3157895	0.734	Alex Caruso	2019-20
18	0.436	6.4	0.3750000	0.645	Alex Caruso	2020-21
19	0.455	5.6	0.3478261	0.808	Alex Caruso	2022-23
20	0.615	6.6	0.2500000	0.629	Alex Len	2020-21
21	0.474	9.6	0.2500000	0.711	Alperen Sengun	2021-22

Showing 1 to 21 of 833 entries, 6 total columns

Let's take a quick precautionary measure and check our data frames for any N/A values which will throw off our **gglot** charts and eventual K-means algorithm. In this case we get back FALSE meaning we don't have any!

any(is.na(wilson_data))

[1] FALSE

any(is.na(spalding_data))

[1] FALSE

Finally, we call the summary function to get inter-quartile range information which shows higher values across for the **wilson_data** dataset which further refutes the players claims of the ball being worse and slipperier. Unless they got accustomed quickly.

#EDA and GGplot graphs

summary(spalding_data$pctFG)

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.3520  0.4248  0.4550  0.4723  0.4983  0.7420
```

summary(wilson_data$pctFG)

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.3580  0.4300  0.4600  0.4767  0.5000  0.7610
```

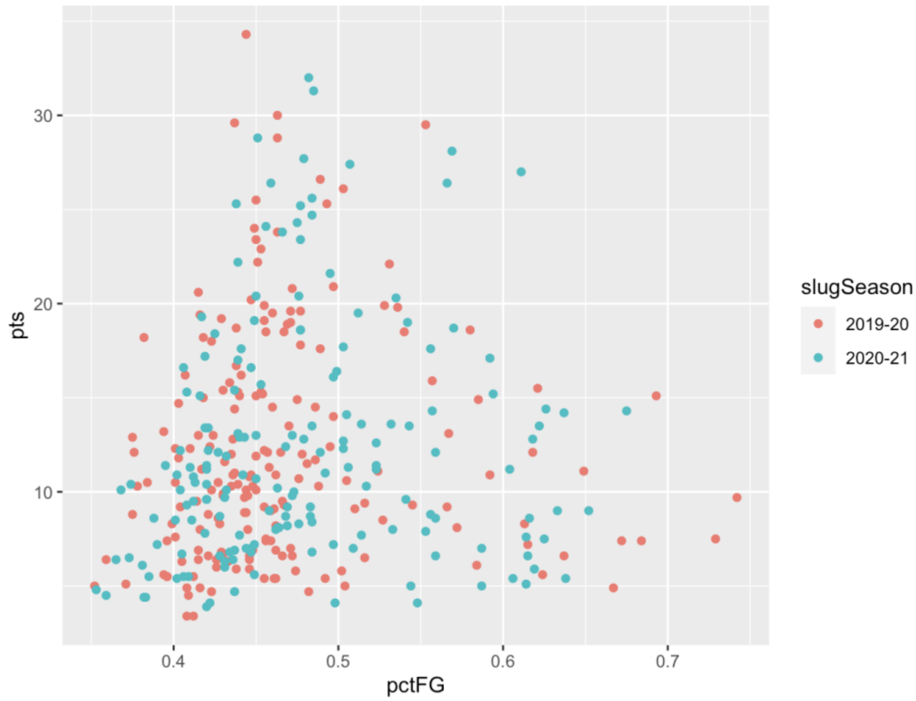
Let's explore this data with charts in Ggplot.

- **dplyr** is an R library for streamlined data manipulation and transformation, providing functions for filtering, grouping, summarizing, and joining datasets, making it essential for data wrangling.
- **ggplot2** is a popular R library for creating customizable data visualizations, allowing users to construct a wide range of plots and easily customize them with colors, themes, and annotations, making it a powerful tool for data visualization in R.

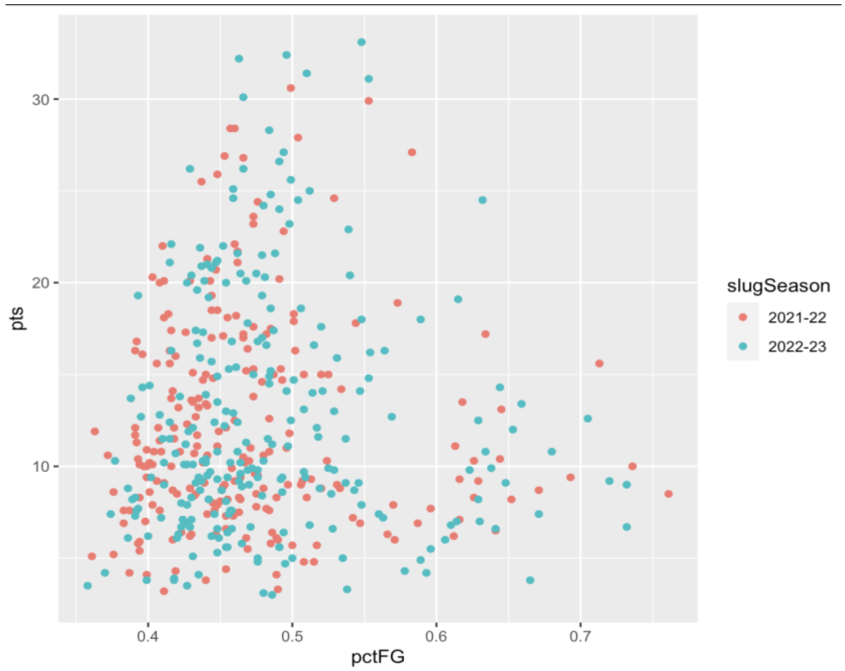
First 2 are a scatter plot labeling field goal percentage by points scored and each dot is labeled for the season in the respective data frame.

```
ggplot(spalding_data, aes(pctFG, pts, color = slugSeason)) + geom_point()
ggplot(wilson_data, aes(pctFG, pts, color = slugSeason)) + geom_point()
```

Spalding:



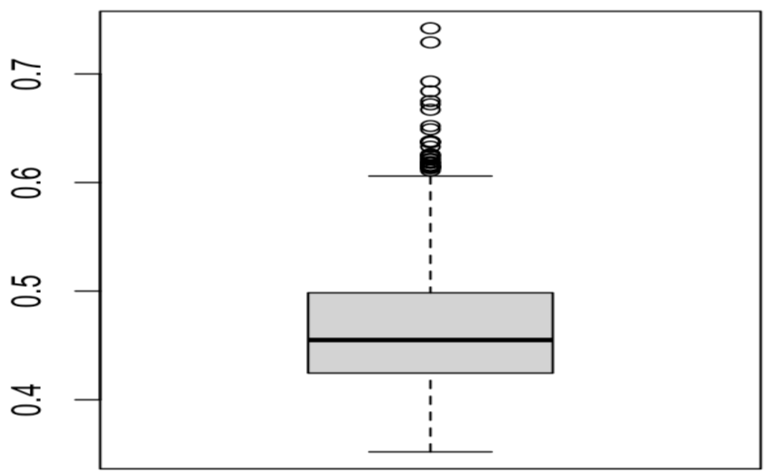
Wilson:



As you can see, if we put this dataframe in a boxplot the boxes and whiskers lines align with the values R Studio calculated.

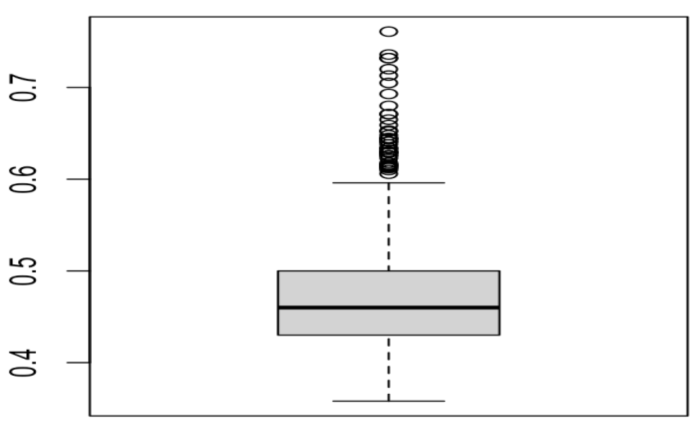
boxplot(spalding_data$pctFG)

Spalding:



boxplot(wilson_data$pctFG)

Wilson:



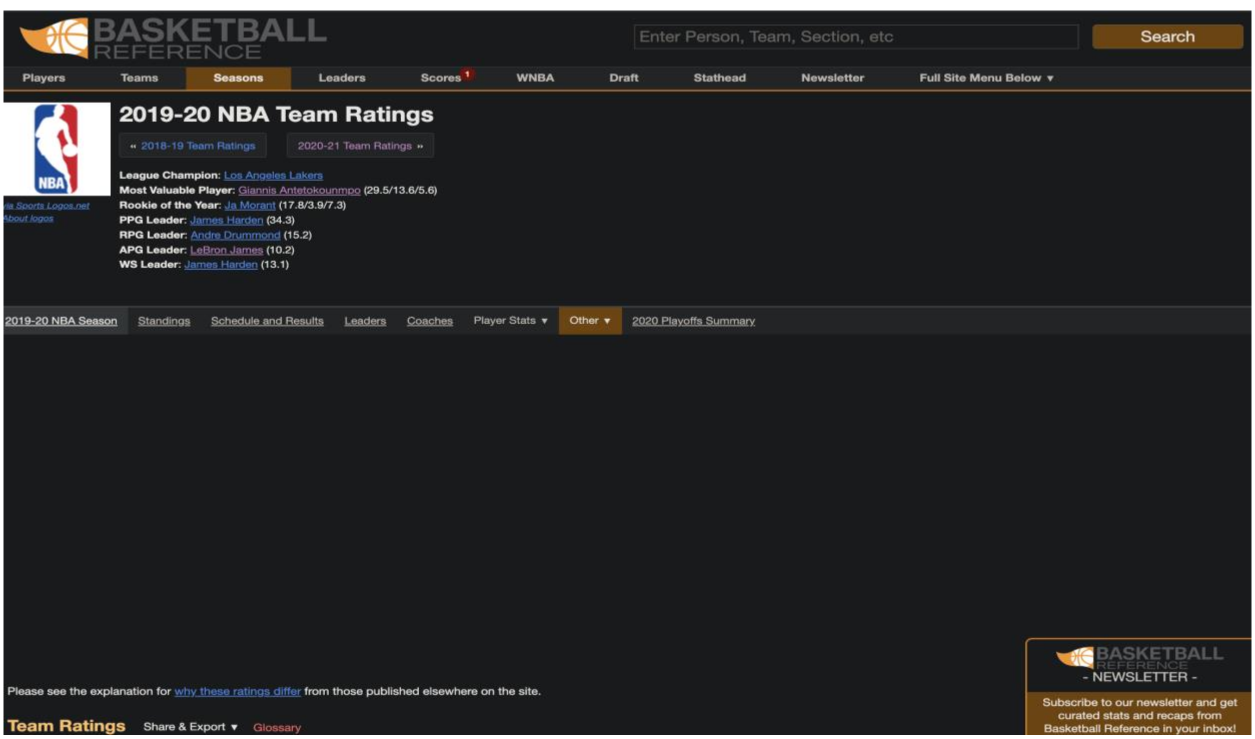
Web Scraping and Advanced Basketball Analytics

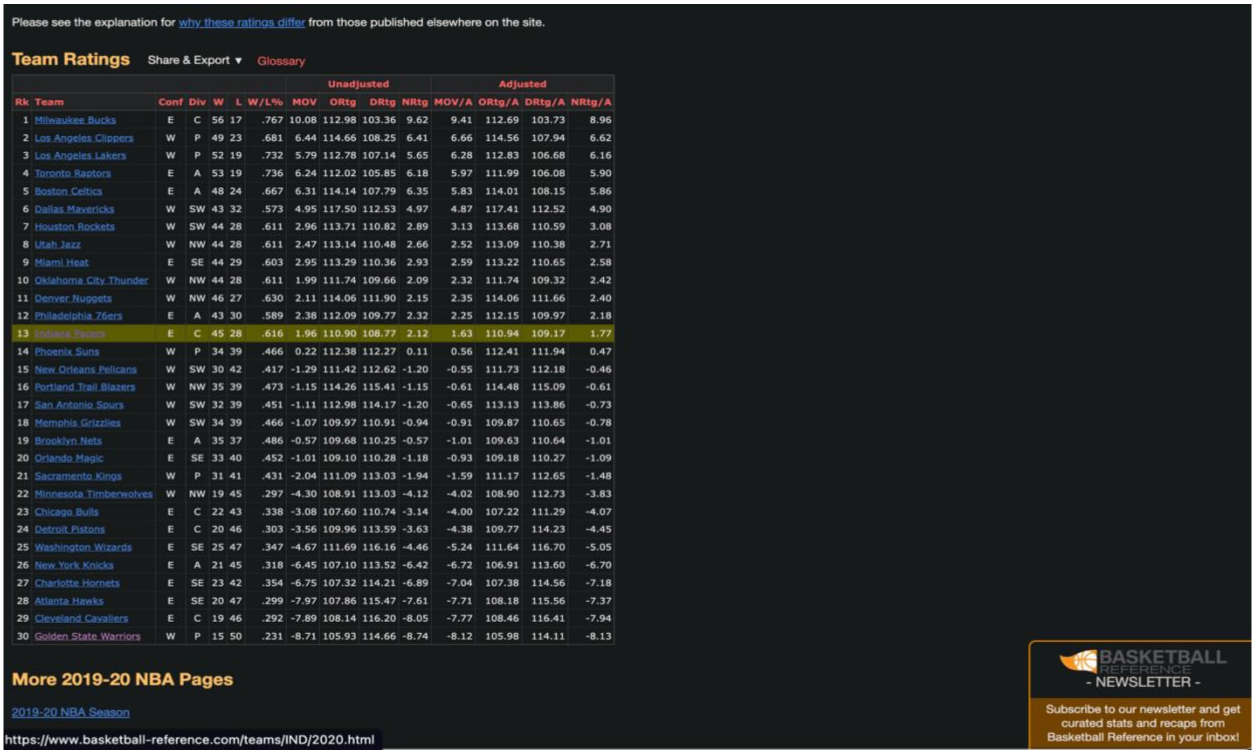
The **NBAStatR** library does not include advanced statistics like Offensive, Defensive and Net Ratings which give us a more comprehensive and concise way of judging cumulative performance. Fortunately, R has a few libraries that help us scrape data from websites by parsing its HTML code. We will be scraping advanced basketball statistics from www.basketballreference.com to gain better insight on if the switch in balls made any noticeable differences anywhere scoring related.

- **rvest** is an R library for web scraping, allowing users to extract data from HTML web pages with functions tailored for navigation and information extraction.
- **stringr** is a versatile R library that simplifies string manipulation tasks, offering intuitive functions for tasks like pattern matching, string splitting, and substitution, making it ideal for data preprocessing and text analysis.
- **reshape** package in R provides functions to transform and restructure data between wide and long formats, facilitating data aggregation, analysis, and visualization, although it has been superseded by the tidyr package in recent versions of R.

```
library(rvest)
library(dplyr)
library(stringr)
library(ggplot2)
library(reshape)
```

First visit the url: https://www.basketball-reference.com/leagues/NBA_2020_ratings.html
Go to the links for each season and scroll down until you see the advanced statistics chart.





Please see the explanation for why these ratings differ from those published elsewhere on the site.

Team Ratings

Share & Export ▾ Glossary

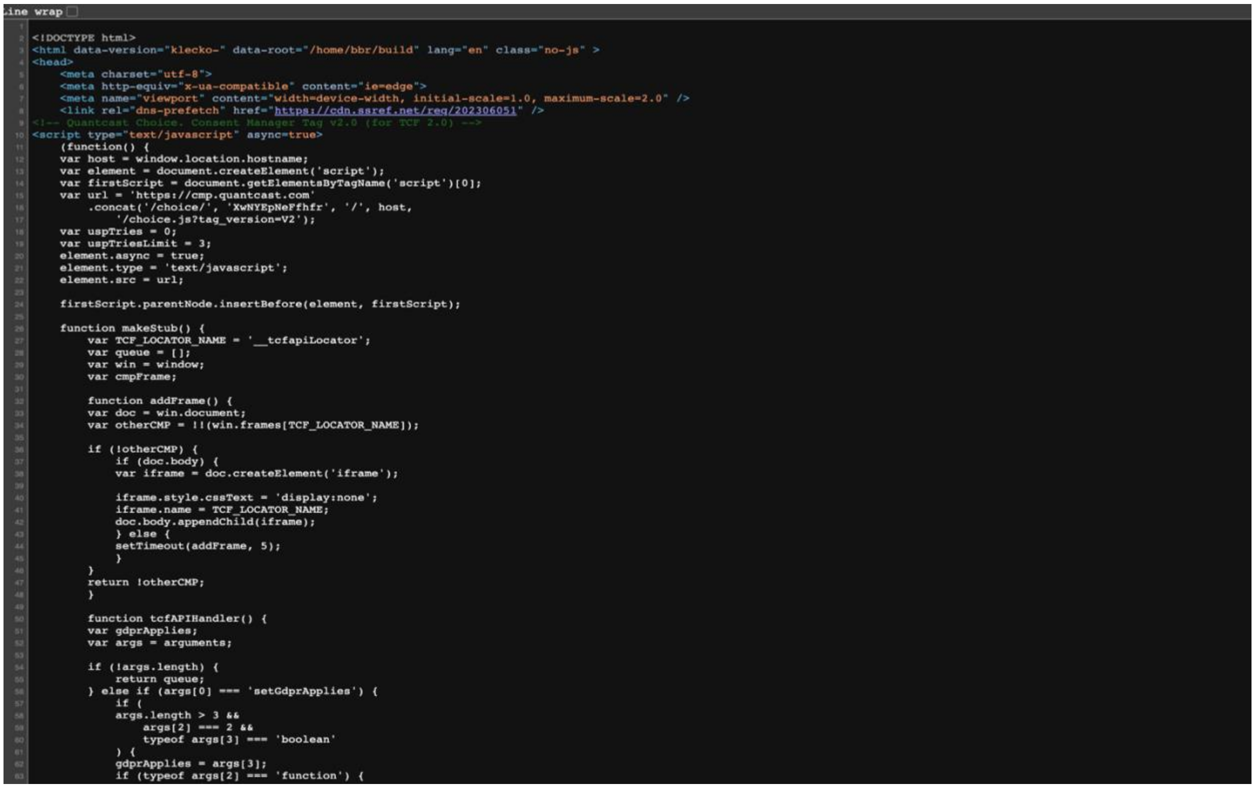
							Unadjusted				Adjusted			
Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	DRtg/A	NRtg/A
1	Milwaukee Bucks	E	C	56	17	.767	10.08	112.98	103.36	9.62	9.41	112.69	103.73	8.96
2	Los Angeles Clippers	W	P	49	23	.681	6.44	114.66	108.25	6.41	6.66	114.56	107.94	6.62
3	Los Angeles Lakers	W	P	52	19	.732	5.79	112.78	107.14	5.65	6.28	112.83	106.68	6.16
4	Toronto Raptors	E	A	53	19	.736	6.24	112.02	105.85	6.18	5.97	111.99	106.08	5.90
5	Boston Celtics	E	A	48	24	.667	6.31	114.14	107.79	6.35	5.83	114.01	108.15	5.86
6	Dallas Mavericks	W	SW	43	32	.573	4.95	117.50	112.53	4.97	4.87	117.41	112.52	4.90
7	Houston Rockets	W	SW	44	28	.611	2.96	113.71	110.82	2.89	3.13	113.68	110.59	3.08
8	Utah Jazz	W	NW	44	28	.611	2.47	113.14	110.48	2.66	2.52	113.09	110.38	2.71
9	Miami Heat	E	SE	44	29	.603	2.95	113.29	110.36	2.93	2.59	113.22	110.65	2.58
10	Oklahoma City Thunder	W	NW	44	28	.611	1.99	111.74	109.66	2.09	2.32	111.74	109.32	2.42
11	Denver Nuggets	W	NW	46	27	.630	2.11	114.06	111.90	2.15	2.35	114.06	111.66	2.40
12	Philadelphia 76ers	E	A	43	30	.589	2.38	112.09	109.77	2.32	2.25	112.15	109.97	2.18
13	Indiana Pacers	E	C	45	28	.616	1.96	110.90	108.77	2.12	1.63	110.94	109.17	1.77
14	Phoenix Suns	W	P	34	39	.466	0.22	112.38	112.27	0.11	0.56	112.41	111.94	0.47
15	New Orleans Pelicans	W	SW	30	42	.417	-1.29	111.42	112.62	-1.20	-0.55	111.73	112.18	-0.46
16	Portland Trail Blazers	W	NW	35	39	.473	-1.15	114.26	115.41	-1.15	-0.61	114.48	115.09	-0.61
17	San Antonio Spurs	W	SW	32	39	.451	-1.11	112.98	114.17	-1.20	-0.65	113.13	113.86	-0.73
18	Memphis Grizzlies	W	SW	34	39	.466	-1.07	109.97	110.91	-0.94	-0.91	109.87	110.65	-0.78
19	Brooklyn Nets	E	A	35	37	.486	-0.57	109.68	110.25	-0.57	-1.01	109.63	110.64	-1.01
20	Orlando Magic	E	SE	33	40	.452	-1.01	109.10	110.28	-1.18	-0.93	109.18	110.27	-1.09
21	Sacramento Kings	W	P	31	41	.431	-2.04	111.09	113.03	-1.94	-1.59	111.17	112.65	-1.48
22	Minnesota Timberwolves	W	NW	19	45	.297	-4.30	108.91	113.03	-4.12	-4.02	108.90	112.73	-3.83
23	Chicago Bulls	E	C	22	43	.338	-3.08	107.60	110.74	-3.14	-4.00	107.22	111.29	-4.07
24	Detroit Pistons	E	C	20	46	.303	-3.56	109.96	113.59	-3.63	-4.38	109.77	114.23	-4.45
25	Washington Wizards	E	SE	25	47	.347	-4.67	111.69	116.16	-4.46	-5.24	111.64	116.70	-5.05
26	New York Knicks	E	A	21	45	.318	-6.45	107.10	113.52	-6.42	-6.72	106.91	113.60	-6.70
27	Charlotte Hornets	E	SE	23	42	.354	-6.75	107.32	114.21	-6.89	-7.04	107.38	114.56	-7.18
28	Atlanta Hawks	E	SE	20	47	.299	-7.97	107.86	115.47	-7.61	-7.71	108.18	115.56	-7.37
29	Cleveland Cavaliers	E	C	19	46	.292	-7.89	108.14	116.20	-8.05	-7.77	108.46	116.41	-7.94
30	Golden State Warriors	W	P	15	50	.231	-8.71	105.93	114.66	-8.74	-8.12	105.98	114.11	-8.13

More 2019-20 NBA Pages

2019-20 NBA Season

https://www.basketball-reference.com/teams/IND/2020.html

Right click on the page and go to Page Source



Line wrap

```
<!DOCTYPE html>
<html data-version="klecko-" data-root="/home/bbr/build" lang="en" class="no-js" >
<head>
    <meta charset="utf-8">
    <meta http-equiv="x-ua-compatible" content="ie=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=2.0" />
    <link rel="dns-prefetch" href="https://cdn.ssref.net/req/202306051" />
<!-- Quantcast Choice. Consent Manager Tag v2.0 (for TCF 2.0) -->
<script type="text/javascript" async=true>
    (function() {
    var host = window.location.hostname;
    var element = document.createElement('script');
    var firstScript = document.getElementsByTagName('script')[0];
    var url = 'https://cmp.quantcast.com'
        .concat('/choice/', 'XwNYEpNeFfhfr', '/', host,
            '/choice.js?tag_version=V2');
    var uspTries = 0;
    var uspTriesLimit = 3;
    element.async = true;
    element.type = 'text/javascript';
    element.src = url;

    firstScript.parentNode.insertBefore(element, firstScript);

    function makeStub() {
        var TCF_LOCATOR_NAME = '__tcfapiLocator';
        var queue = [];
        var win = window;
        var cmpFrame;

        function addFrame() {
        var doc = win.document;
        var otherCMP = !!(win.frames[TCF_LOCATOR_NAME]);

        if (!otherCMP) {
            if (doc.body) {
            var iframe = doc.createElement('iframe');

            iframe.style.cssText = 'display:none';
            iframe.name = TCF_LOCATOR_NAME;
            doc.body.appendChild(iframe);
            } else {
            setTimeout(addFrame, 5);
            }
        }
        return !otherCMP;
        }

        function tcfAPIHandler() {
        var gdprApplies;
        var args = arguments;

        if (!args.length) {
            return queue;
        } else if (args[0] === 'setGdprApplies') {
            if (
            args.length > 3 &&
                args[2] === 2 &&
                typeof args[3] === 'boolean'
            ) {
            gdprApplies = args[3];
            if (typeof args[2] === 'function') {
```

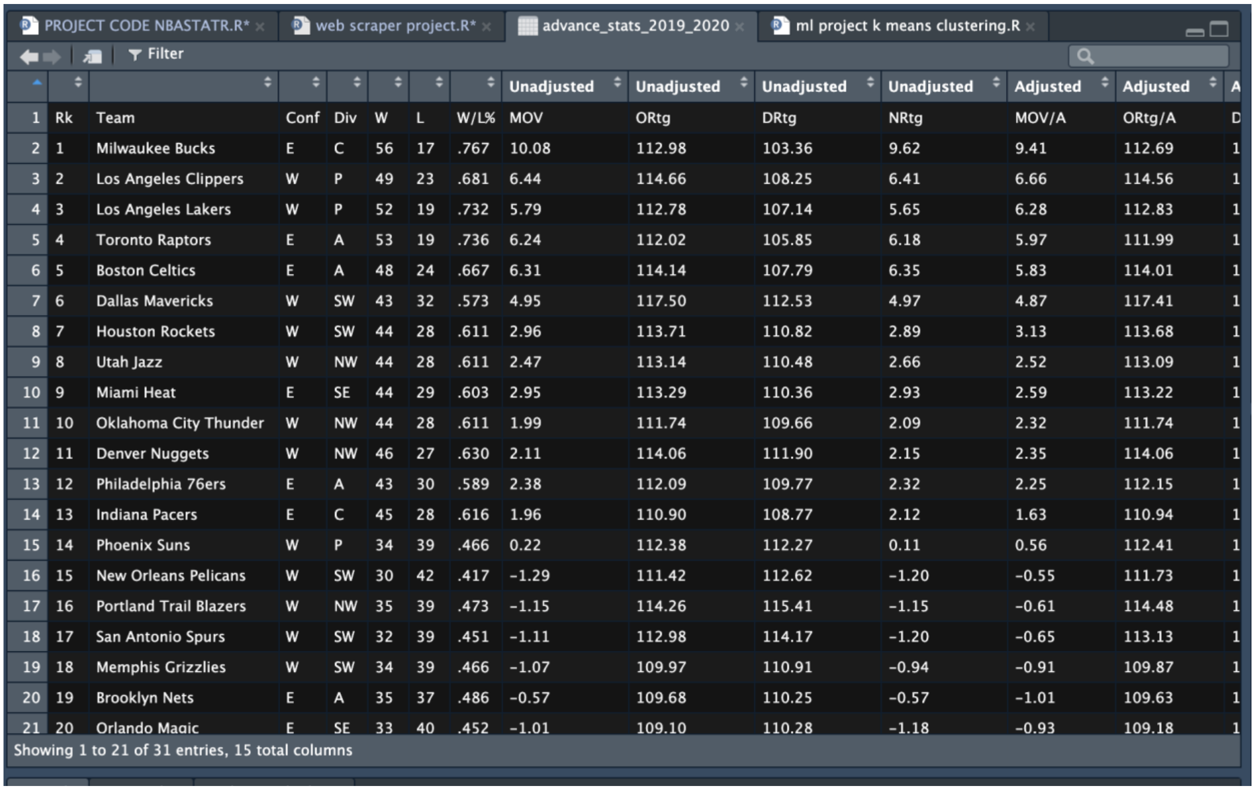
Press control + F (command + F in Mac) and type in “Table”.

The <Table> tags in HTML are essential for web scraping with R as they structure tabular data on web pages. They serve as markers for identifying and extracting specific data during web scraping tasks. R libraries like **rvest** utilize these tags to locate and extract tables efficiently. Although some websites are built to be difficult to scrape and parsing through a screen of HTML markup might reveal other tables within the page. Basketball Reference is fortunately an easy website to scrape since the tables are clearly distinct from one another. In the above image this is the first instance of the table tag being used so we will assume it’s the first table with our data.

```
# 2019-20 web scraper

table_url <- read_html('https://www.basketball-reference.com/leagues/NBA_2020_ratings.html')

advance_stats_2019_2020 <- table_url %>% html_node('table') %>% html_table(fill = TRUE)
```



								Unadjusted	Unadjusted	Unadjusted	Unadjusted	Adjusted	Adjusted	A
1	Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	D
2	1	Milwaukee Bucks	E	C	56	17	.767	10.08	112.98	103.36	9.62	9.41	112.69	1
3	2	Los Angeles Clippers	W	P	49	23	.681	6.44	114.66	108.25	6.41	6.66	114.56	1
4	3	Los Angeles Lakers	W	P	52	19	.732	5.79	112.78	107.14	5.65	6.28	112.83	1
5	4	Toronto Raptors	E	A	53	19	.736	6.24	112.02	105.85	6.18	5.97	111.99	1
6	5	Boston Celtics	E	A	48	24	.667	6.31	114.14	107.79	6.35	5.83	114.01	1
7	6	Dallas Mavericks	W	SW	43	32	.573	4.95	117.50	112.53	4.97	4.87	117.41	1
8	7	Houston Rockets	W	SW	44	28	.611	2.96	113.71	110.82	2.89	3.13	113.68	1
9	8	Utah Jazz	W	NW	44	28	.611	2.47	113.14	110.48	2.66	2.52	113.09	1
10	9	Miami Heat	E	SE	44	29	.603	2.95	113.29	110.36	2.93	2.59	113.22	1
11	10	Oklahoma City Thunder	W	NW	44	28	.611	1.99	111.74	109.66	2.09	2.32	111.74	1
12	11	Denver Nuggets	W	NW	46	27	.630	2.11	114.06	111.90	2.15	2.35	114.06	1
13	12	Philadelphia 76ers	E	A	43	30	.589	2.38	112.09	109.77	2.32	2.25	112.15	1
14	13	Indiana Pacers	E	C	45	28	.616	1.96	110.90	108.77	2.12	1.63	110.94	1
15	14	Phoenix Suns	W	P	34	39	.466	0.22	112.38	112.27	0.11	0.56	112.41	1
16	15	New Orleans Pelicans	W	SW	30	42	.417	−1.29	111.42	112.62	−1.20	−0.55	111.73	1
17	16	Portland Trail Blazers	W	NW	35	39	.473	−1.15	114.26	115.41	−1.15	−0.61	114.48	1
18	17	San Antonio Spurs	W	SW	32	39	.451	−1.11	112.98	114.17	−1.20	−0.65	113.13	1
19	18	Memphis Grizzlies	W	SW	34	39	.466	−1.07	109.97	110.91	−0.94	−0.91	109.87	1
20	19	Brooklyn Nets	E	A	35	37	.486	−0.57	109.68	110.25	−0.57	−1.01	109.63	1
21	20	Orlando Magic	E	SE	33	40	.452	−1.01	109.10	110.28	−1.18	−0.93	109.18	1

Success! We successfully scraped the data from the web and have it in a nice tabular shape. We will need to clean up the first row which R is reading as the label row.

advance_stats_2019_2020 <- advance_stats_2019_2020

colnames(advance_stats_2019_2020) <- advance_stats_2019_2020 [1,]

advance_stats_2019_2020

advance_stats_2019_2020 <- advance_stats_2019_2020 [-1,]

PROJECT CODE NBASTATR.R* | web scraper project.R* | advance_stats_2019_2020 | ml project k means clustering.R

Filter

	Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	DRtg/A
1	Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	DRtg/A
2	1	Milwaukee Bucks	E	C	56	17	.767	10.08	112.98	103.36	9.62	9.41	112.69	103.73
3	2	Los Angeles Cl Milwaukee Bucks		P	49	23	.681	6.44	114.66	108.25	6.41	6.66	114.56	107.94
4	3	Los Angeles Lakers	W	P	52	19	.732	5.79	112.78	107.14	5.65	6.28	112.83	106.68
5	4	Toronto Raptors	E	A	53	19	.736	6.24	112.02	105.85	6.18	5.97	111.99	106.08
6	5	Boston Celtics	E	A	48	24	.667	6.31	114.14	107.79	6.35	5.83	114.01	108.15
7	6	Dallas Mavericks	W	SW	43	32	.573	4.95	117.50	112.53	4.97	4.87	117.41	112.52
8	7	Houston Rockets	W	SW	44	28	.611	2.96	113.71	110.82	2.89	3.13	113.68	110.59
9	8	Utah Jazz	W	NW	44	28	.611	2.47	113.14	110.48	2.66	2.52	113.09	110.38
10	9	Miami Heat	E	SE	44	29	.603	2.95	113.29	110.36	2.93	2.59	113.22	110.65
11	10	Oklahoma City Thunder	W	NW	44	28	.611	1.99	111.74	109.66	2.09	2.32	111.74	109.32
12	11	Denver Nuggets	W	NW	46	27	.630	2.11	114.06	111.90	2.15	2.35	114.06	111.66
13	12	Philadelphia 76ers	E	A	43	30	.589	2.38	112.09	109.77	2.32	2.25	112.15	109.97
14	13	Indiana Pacers	E	C	45	28	.616	1.96	110.90	108.77	2.12	1.63	110.94	109.17
15	14	Phoenix Suns	W	P	34	39	.466	0.22	112.38	112.27	0.11	0.56	112.41	111.94
16	15	New Orleans Pelicans	W	SW	30	42	.417	−1.29	111.42	112.62	−1.20	−0.55	111.73	112.18
17	16	Portland Trail Blazers	W	NW	35	39	.473	−1.15	114.26	115.41	−1.15	−0.61	114.48	115.09
18	17	San Antonio Spurs	W	SW	32	39	.451	−1.11	112.98	114.17	−1.20	−0.65	113.13	113.86
19	18	Memphis Grizzlies	W	SW	34	39	.466	−1.07	109.97	110.91	−0.94	−0.91	109.87	110.65
20	19	Brooklyn Nets	E	A	35	37	.486	−0.57	109.68	110.25	−0.57	−1.01	109.63	110.64
21	20	Orlando Magic	E	SE	33	40	.452	−1.01	109.10	110.28	−1.18	−0.93	109.18	110.27

Showing 1 to 21 of 31 entries, 15 total columns

Here we are going to append a new column Season for each of the 4 data frames. We need that column so when we go to combine the data frames, we can easily see what season each row is a part of.

```
Season = c('2019-2020 Season')
advance_stats_2019_2020$Season <- Season
advance_stats_2019_2020
```

PROJECT CODE NBASTATR.R* | web scraper project.R* | advance_stats_2019_2020 | ml project k means clustering.R

Filter

	Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	DRtg/A	NRtg/A	Season
1	1	Milwaukee Bucks	E	C	56	17	.767	10.08	112.98	103.36	9.62	9.41	112.69	103.73	8.96	2019-2020 Season
2	2	Los Angeles Clippers	W	P	49	23	.681	6.44	114.66	108.25	6.41	6.66	114.56	107.94	6.62	2019-2020 Season
3	3	Los Angeles Lakers	W	P	52	19	.732	5.79	112.78	107.14	5.65	6.28	112.83	106.68	6.16	2019-2020 Season
4	4	Toronto Raptors	E	A	53	19	.736	6.24	112.02	105.85	6.18	5.97	111.99	106.08	5.90	2019-2020 Season
5	5	Boston Celtics	E	A	48	24	.667	6.31	114.14	107.79	6.35	5.83	114.01	108.15	5.86	2019-2020 Season
6	6	Dallas Mavericks	W	SW	43	32	.573	4.95	117.50	112.53	4.97	4.87	117.41	112.52	4. 5.86	2019-2020 Season
7	7	Houston Rockets	W	SW	44	28	.611	2.96	113.71	110.82	2.89	3.13	113.68	110.59	3.08	2019-2020 Season
8	8	Utah Jazz	W	NW	44	28	.611	2.47	113.14	110.48	2.66	2.52	113.09	110.38	2.71	2019-2020 Season
9	9	Miami Heat	E	SE	44	29	.603	2.95	113.29	110.36	2.93	2.59	113.22	110.65	2.58	2019-2020 Season
10	10	Oklahoma City Thunder	W	NW	44	28	.611	1.99	111.74	109.66	2.09	2.32	111.74	109.32	2.42	2019-2020 Season
11	11	Denver Nuggets	W	NW	46	27	.630	2.11	114.06	111.90	2.15	2.35	114.06	111.66	2.40	2019-2020 Season
12	12	Philadelphia 76ers	E	A	43	30	.589	2.38	112.09	109.77	2.32	2.25	112.15	109.97	2.18	2019-2020 Season
13	13	Indiana Pacers	E	C	45	28	.616	1.96	110.90	108.77	2.12	1.63	110.94	109.17	1.77	2019-2020 Season
14	14	Phoenix Suns	W	P	34	39	.466	0.22	112.38	112.27	0.11	0.56	112.41	111.94	0.47	2019-2020 Season
15	15	New Orleans Pelicans	W	SW	30	42	.417	−1.29	111.42	112.62	−1.20	−0.55	111.73	112.18	−0.46	2019-2020 Season
16	16	Portland Trail Blazers	W	NW	35	39	.473	−1.15	114.26	115.41	−1.15	−0.61	114.48	115.09	−0.61	2019-2020 Season
17	17	San Antonio Spurs	W	SW	32	39	.451	−1.11	112.98	114.17	−1.20	−0.65	113.13	113.86	−0.73	2019-2020 Season
18	18	Memphis Grizzlies	W	SW	34	39	.466	−1.07	109.97	110.91	−0.94	−0.91	109.87	110.65	−0.78	2019-2020 Season
19	19	Brooklyn Nets	E	A	35	37	.486	−0.57	109.68	110.25	−0.57	−1.01	109.63	110.64	−1.01	2019-2020 Season
20	20	Orlando Magic	E	SE	33	40	.452	−1.01	109.10	110.28	−1.18	−0.93	109.18	110.27	−1.09	2019-2020 Season
21	21	Sacramento Kings	W	P	31	41	.431	−2.04	111.09	113.03	−1.94	−1.59	111.17	112.65	−1.48	2019-2020 Season

Showing 1 to 21 of 30 entries, 16 total columns

This is the rest of the code for each other seasons data frame:

```
# 2020-21 web scraper

table_url <- read_html('https://www.basketball-reference.com/leagues/NBA_2021_ratings.html')

advance_stats_2020_2021 <- table_url %>% html_node('table') %>% html_table(fill = TRUE)

####

advance_stats_2020_2021 <- advance_stats_2020_2021

####

colnames(advance_stats_2020_2021 ) <- advance_stats_2020_2021 [1,]

advance_stats_2020_2021

advance_stats_2020_2021  <- advance_stats_2020_2021 [-1,]

# adding column that says season year


Season = c('2020-2021 Season')
advance_stats_2020_2021$Season <- Season
advance_stats_2020_2021




# 2021-22 web scraper

table_url <- read_html('https://www.basketball-reference.com/leagues/NBA_2022_ratings.html')

advance_stats_2021_2022 <- table_url %>% html_node('table') %>% html_table(fill = TRUE)



advance_stats_2021_2022 <- advance_stats_2021_2022
```

```
colnames(advance_stats_2021_2022 ) <- advance_stats_2021_2022 [1,]

advance_stats_2021_2022

advance_stats_2021_2022  <- advance_stats_2021_2022 [-1,]


# adding column that says season year

Season = c('2021-2022 Season')
advance_stats_2021_2022$Season <- Season
advance_stats_2021_2022




# 2022-23 web scraper

table_url <- read_html('https://www.basketball-
reference.com/leagues/NBA_2023_ratings.html')

advance_stats_2022_2023 <- table_url %>% html_node('table') %>% html_table(fill =
TRUE)



advance_stats_2022_2023 <- advance_stats_2022_2023



colnames(advance_stats_2022_2023 ) <- advance_stats_2022_2023 [1,]

advance_stats_2022_2023

advance_stats_2022_2023  <- advance_stats_2022_2023 [-1,]


# adding column that says season year

Season = c('2022-2023 Season')
advance_stats_2022_2023$Season <- Season
advance_stats_2022_2023
```

Quickly convert the Offensive Rating (ORtg) column in each data frame to numeric since it is a number value that isn't already typed as one.

In basketball, offensive rating is a statistic used to measure the efficiency of an individual player, team, or lineup in scoring points. It quantifies the number of points a player or team produces per 100 possessions. Offensive rating considers factors such as field goals made, free throws made, turnovers, and offensive rebounds.

A higher offensive rating indicates a more efficient offense, as it means that the player, team, or lineup is generating a higher number of points per possession. It is a useful metric for evaluating offensive performance and comparing players or teams in terms of scoring efficiency. Offensive rating can be influenced by various factors, including shooting accuracy, shot selection, passing ability, and overall offensive strategy. It provides valuable insights into the effectiveness and productivity of an offensive unit in generating points and can be used to identify efficient offensive players or teams within the context of the NBA or any basketball league.

```
advance_stats_2019_2020$ORtg <-
as.numeric(as.character(advance_stats_2019_2020$ORtg))

advance_stats_2020_2021$ORtg <-
as.numeric(as.character(advance_stats_2020_2021$ORtg))

advance_stats_2021_2022$ORtg <-
as.numeric(as.character(advance_stats_2021_2022$ORtg))

advance_stats_2022_2023$ORtg <-
as.numeric(as.character(advance_stats_2022_2023$ORtg))
```

Now let's find the average of that same Offensive Rating column for each data frame. Looks like 2019-2020 came up to about 111.3, 2020-2021 at 113.1, then 112.9, and lastly 115.7.

```
mean(advance_stats_2019_2020$ORtg)
[1] 111.28

mean(advance_stats_2020_2021$ORtg)
[1] 113.1007

mean(advance_stats_2021_2022$ORtg)
[1] 112.8513

mean(advance_stats_2022_2023$ORtg)
[1] 115.6827
```

The jump from 111.3 in 2019-2020 to 115.7 in 2022-2023 is interesting but still does not support the claim that the Wilson balls hindered scoring performance. The last Spalding season and Wilson season are at 113.1 and 112.9 respectively which comes down to a difference of .2 which is negligible to say the least especially considering that the prior off season was shortened due to COVID in 2020-2021. Now for good measure let's look at Defensive Rating (DRtg) and Net Rating (NRtg).

Defensive rating is a statistic used to measure the effectiveness of an individual player, team, or lineup in preventing the opposing team from scoring. It represents the number of points allowed per 100 possessions. A lower defensive rating indicates a more effective defense, as it suggests that the player, team, or lineup is giving up fewer points per possession. Defensive rating considers factors such as field goals allowed, free throws allowed, turnovers forced, and defensive rebounds.

Net rating, on the other hand, provides an overall measure of a player, team, or lineup's performance by considering both offensive and defensive ratings. It is calculated by subtracting the defensive rating from the offensive rating. Net rating reflects the point differential per 100 possessions, indicating how much a player, team, or lineup outperforms or underperforms their opponents on average. A positive net rating suggests a better overall performance, while a negative net rating indicates a performance below the opponents' average.

```
advance_stats_2019_2020$DRtg <-
as.numeric(as.character(advance_stats_2019_2020$DRtg))
advance_stats_2020_2021$DRtg <-
as.numeric(as.character(advance_stats_2020_2021$DRtg))
advance_stats_2021_2022$DRtg <-
as.numeric(as.character(advance_stats_2021_2022$DRtg))
advance_stats_2022_2023$DRtg <-
as.numeric(as.character(advance_stats_2022_2023$DRtg))
```

```
mean(advance_stats_2019_2020$DRtg)
[1] 111.44
mean(advance_stats_2020_2021$DRtg)
[1] 113.1007
mean(advance_stats_2021_2022$DRtg)
[1] 112.851
mean(advance_stats_2022_2023$DRtg)
[1] 115.6827
```

Now this is very interesting. It looks like again the Spalding final season and Wilson first season have close enough defensive rating means to call a truce but the 2019-2022 has the lowest rating at 111.4 and 2022-2023 at 115.7 which means defense is trending upwards along with offense. This is very counter intuitive since naturally as scoring improvement comes at the detriment of the defensive player. Unless the game is getting so quick that players can jack up enough efficient good shots on aggregate to overcompensate for the defenses getting better.

Same for Net Rating:

```
advance_stats_2019_2020$NRtg <-
as.numeric(as.character(advance_stats_2019_2020$NRtg))
advance_stats_2020_2021$NRtg <-
as.numeric(as.character(advance_stats_2020_2021$NRtg))
advance_stats_2021_2022$NRtg <-
as.numeric(as.character(advance_stats_2021_2022$NRtg))
advance_stats_2022_2023$NRtg <-
as.numeric(as.character(advance_stats_2022_2023$NRtg))
```

```
mean(advance_stats_2019_2020$NRtg)
[1] -0.1596667
mean(advance_stats_2020_2021$NRtg)
[1] -0.0003333333
mean(advance_stats_2021_2022$NRtg)
[1] -0.0003333333
mean(advance_stats_2022_2023$NRtg)
[1] 1.532339e-17
```

Looks like teams in the 2022-2023 season on average had a positive net rating while the previous 3 seasons had all negatives. Still slightly supporting (emphasis on slightly) the opposite narrative around the Spalding balls being a negative for scoring.

Next, we will combine the 4 advanced statistics data frames into 1 large data frame using **rbind**. Notice the Season column which we created and appended earlier.

```
############# Combining the 4 seasons advanced stats ###############

combined.advanced.stats <-
rbind(advance_stats_2019_2020,advance_stats_2020_2021,advance_stats_2021_2022,advance_stats_2022_2023)
```

PROJECT CODE NBASTATR.R* | web scraper project.R* | combined.advanced.stats | advance_stats_2021_2022 | advance_stats_2019_2020 | ml project k means clustering

Filter

	Rk	Team	Conf	Div	W	L	W/L%	MOV	ORtg	DRtg	NRtg	MOV/A	ORtg/A	DRtg/A	NRtg/A	Season
1	19	San Antonio Spurs	W	SW	34	48	.415	0.12	113.04	113.00	0.03	0.02	113.09	113.16	−0.07	2021–2022 Season
2	17	Indiana Pacers	E	C	34	38	.472	−0.04	112.98	112.96	0.02	−0.12	112.99	113.07	−0.08	2020–2021 Season
3	19	Dallas Mavericks	W	SW	38	44	.463	0.07	117.63	117.56	0.06	−0.14	117.54	117.70	−0.16	2022–2023 Season
4	20	Minnesota Timberwolves	W	NW	42	40	.512	−0.04	114.52	114.50	0.02	−0.22	114.26	114.43	−0.18	2022–2023 Season
5	18	New Orleans Pelicans	W	SW	31	41	.431	−0.29	114.47	114.78	−0.32	−0.20	114.52	114.73	−0.21	2020–2021 Season
6	21	Miami Heat	E	SE	44	38	.537	−0.32	113.99	114.40	−0.41	−0.13	114.10	114.32	−0.22	2022–2023 Season
7	20	Chicago Bulls	E	C	46	36	.561	−0.39	114.07	114.55	−0.48	−0.37	114.13	114.58	−0.45	2021–2022 Season
8	15	New Orleans Pelicans	W	SW	30	42	.417	−1.29	111.42	112.62	−1.20	−0.55	111.73	112.18	−0.46	2019–2020 Season
9	19	Toronto Raptors	E	A	27	45	.375	−0.47	112.96	113.44	−0.48	−0.54	113.00	113.55	−0.56	2020–2021 Season
10	16	Portland Trail Blazers	W	NW	35	39	.473	−1.15	114.26	115.41	−1.15	−0.61	114.48	115.09	−0.61	2019–2020 Season
11	17	San Antonio Spurs	W	SW	32	39	.451	−1.11	112.98	114.17	−1.20	−0.65	113.13	113.86	−0.73	2019–2020 Season
12	18	Memphis Grizzlies	W	SW	34	39	.466	−1.07	109.97	110.91	−0.94	−0.91	109.87	110.65	−0.78	2019–2020 Season
13	21	New Orleans Pelicans	W	SW	36	46	.439	−0.99	112.84	113.82	−0.98	−0.84	112.93	113.77	−0.84	2021–2022 Season
14	20	Chicago Bulls	E	C	31	41	.431	−0.89	111.90	112.81	−0.91	−0.94	111.90	112.87	−0.97	2020–2021 Season
15	19	Brooklyn Nets	E	A	35	37	.486	−0.57	109.68	110.25	−0.57	−1.01	109.63	110.64	−1.01	2019–2020 Season
16	20	Orlando Magic	E	SE	33	40	.452	−1.01	109.10	110.28	−1.18	−0.93	109.18	110.27	−1.09	2019–2020 Season
17	22	Utah Jazz	W	NW	37	45	.451	−0.94	116.41	117.44	−1.02	−1.03	116.32	117.43	−1.11	2022–2023 Season
18	23	Washington Wizards	E	SE	35	47	.427	−1.21	115.21	116.49	−1.28	−1.06	115.39	116.50	−1.12	2022–2023 Season
19	21	Sacramento Kings	W	P	31	41	.431	−2.04	111.09	113.03	−1.94	−1.59	111.17	112.65	−1.48	2019–2020 Season
20	21	San Antonio Spurs	W	SW	33	39	.458	−1.74	111.61	113.35	−1.74	−1.58	111.59	113.17	−1.57	2020–2021 Season
21	22	Washington Wizards	E	SE	34	38	.472	−1.83	111.75	113.63	−1.88	−1.84	111.79	113.69	−1.91	2020–2021 Season

Showing 1 to 21 of 120 entries, 16 total columns

Console | Terminal | Background Jobs

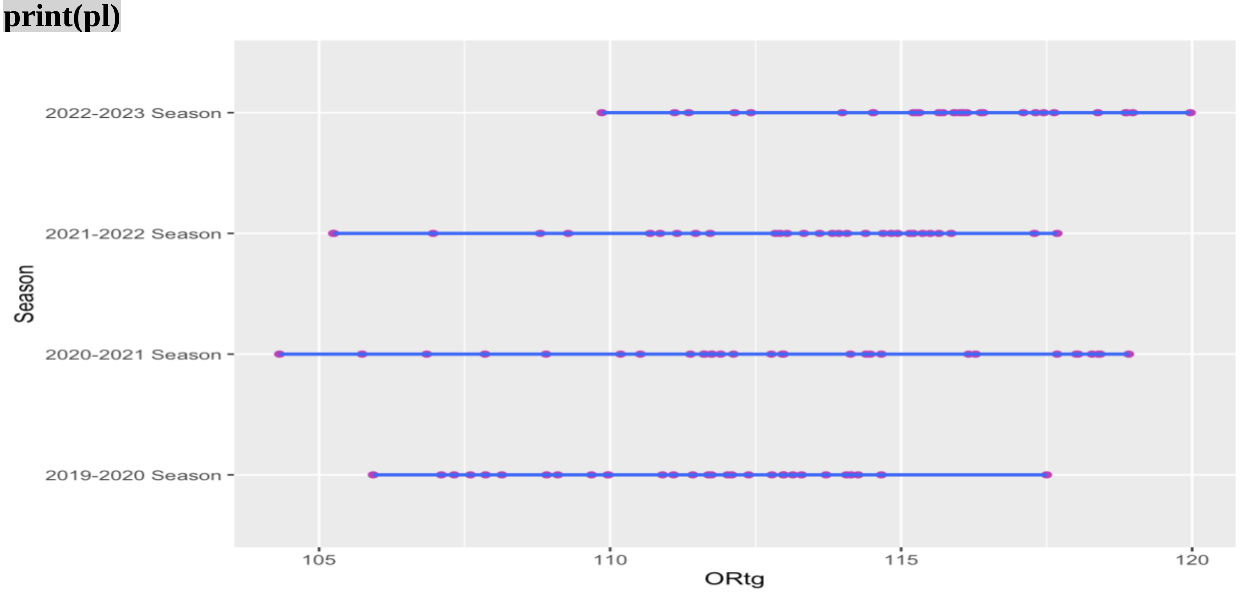
This simple point plot easily illustrates the number of teams within the 4 seasons between 2019-2023 and how high/low the ORtg rating was compared to other instances representing teams in the same season and points on different lines representing teams of those seasons.

We can see here that the 2022-2023 has a slightly higher center cluster and the highest value on the entire chart. Then we see 2020-2021 and 2021-2022 neck and neck as we are used to seeing so far and then 2019-2020 which is very close to the former 2 but doesn't have as large a range.

##################### plot of the offensive rating for each season ###################

```
pl <-ggplot(combined.advanced.stats,aes(x = ORtg, y = Season)) +
        geom_point(color = Season, alpha = 4) + geom_smooth(method=lm)

print(pl)
```

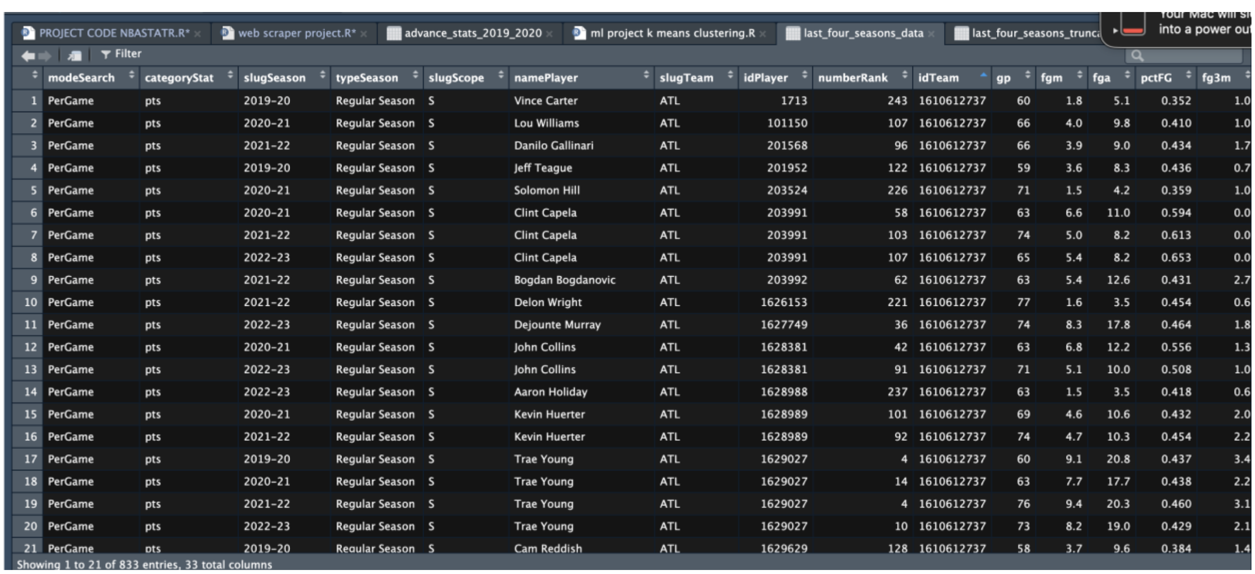


K-Means Clustering

K-means clustering is a popular unsupervised machine learning algorithm that aims to group similar data points together into distinct clusters. The algorithm starts by randomly selecting K cluster centroids and assigns each data point to the nearest centroid based on their distance. Then, it iteratively updates the centroids by calculating the mean position of the data points assigned to each cluster. This process continues until the centroids stabilize, resulting in well-defined clusters. The goal of K-means clustering is to minimize the within-cluster variance, ensuring that data points within the same cluster are as similar as possible, while different clusters are as dissimilar as possible. It is widely used in various fields for tasks such as customer segmentation, image compression, and anomaly detection.

For our purposes we will feed an unlabeled data set into the K-means algorithm with a row for each player season for the 4 seasons of 2019-2020, 2020-2021, 2021-2022, 2022-2023. Only showing the model offensive production data: pctFG, pts, pctFG3, and pctFG. We will then break the cluster into 4 groups and see if the cluster patterns resemble our preconceived notion that the 4 seasons have not shown any drastic scoring dips and there is a slight positive trend in scoring totals and efficiency since Wilson took over basketball manufacturing.

First, we will use a data frame we created earlier called **last_four_seasons_data** which combines our 4 season data frames we created from the **NBAStatR** library. Next, we will put that data on a scatter plot to see what it looks like and we will set the aesthetic to pctFG and color the data points by season (we set the size to 4 but you can change it depending on your viewing needs)



PROJECT CODE NBASTATR.R* | web scraper project.R* | advance_stats_2019_2020 | ml project k means clustering.R | last_four_seasons_data | last_four_seasons_trunca

Filter

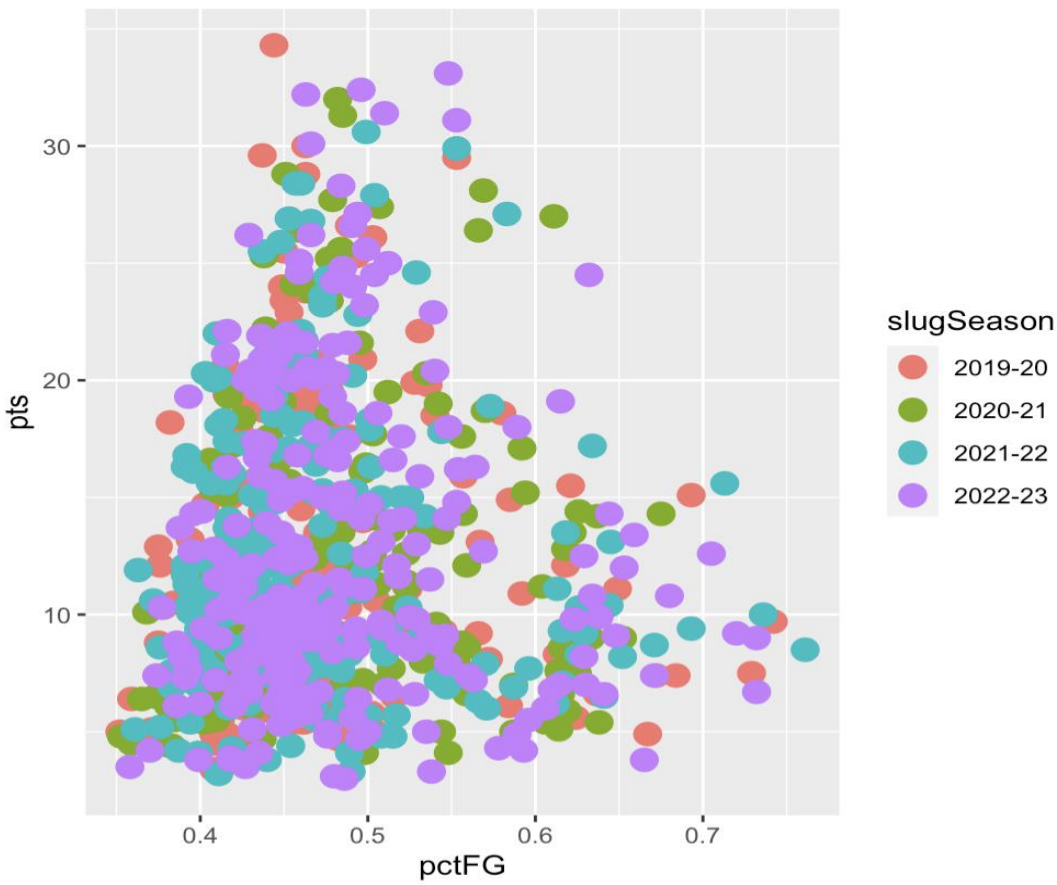
	modeSearch	categoryStat	slugSeason	typeSeason	slugScope	namePlayer	slugTeam	idPlayer	numberRank	idTeam	gp	fgm	fga	pctFG	fg3m
1	PerGame	pts	2019-20	Regular Season	S	Vince Carter	ATL	1713	243	1610612737	60	1.8	5.1	0.352	1.0
2	PerGame	pts	2020-21	Regular Season	S	Lou Williams	ATL	101150	107	1610612737	66	4.0	9.8	0.410	1.0
3	PerGame	pts	2021-22	Regular Season	S	Danilo Gallinari	ATL	201568	96	1610612737	66	3.9	9.0	0.434	1.7
4	PerGame	pts	2019-20	Regular Season	S	Jeff Teague	ATL	201952	122	1610612737	59	3.6	8.3	0.436	0.7
5	PerGame	pts	2020-21	Regular Season	S	Solomon Hill	ATL	203524	226	1610612737	71	1.5	4.2	0.359	1.0
6	PerGame	pts	2020-21	Regular Season	S	Clint Capela	ATL	203991	58	1610612737	63	6.6	11.0	0.594	0.0
7	PerGame	pts	2021-22	Regular Season	S	Clint Capela	ATL	203991	103	1610612737	74	5.0	8.2	0.613	0.0
8	PerGame	pts	2022-23	Regular Season	S	Clint Capela	ATL	203991	107	1610612737	65	5.4	8.2	0.653	0.0
9	PerGame	pts	2021-22	Regular Season	S	Bogdan Bogdanovic	ATL	203992	62	1610612737	63	5.4	12.6	0.431	2.7
10	PerGame	pts	2021-22	Regular Season	S	Delon Wright	ATL	1626153	221	1610612737	77	1.6	3.5	0.454	0.6
11	PerGame	pts	2022-23	Regular Season	S	Dejounte Murray	ATL	1627749	36	1610612737	74	8.3	17.8	0.464	1.8
12	PerGame	pts	2020-21	Regular Season	S	John Collins	ATL	1628381	42	1610612737	63	6.8	12.2	0.556	1.3
13	PerGame	pts	2022-23	Regular Season	S	John Collins	ATL	1628381	91	1610612737	71	5.1	10.0	0.508	1.0
14	PerGame	pts	2022-23	Regular Season	S	Aaron Holiday	ATL	1628988	237	1610612737	63	1.5	3.5	0.418	0.6
15	PerGame	pts	2020-21	Regular Season	S	Kevin Huerter	ATL	1628989	101	1610612737	69	4.6	10.6	0.432	2.0
16	PerGame	pts	2021-22	Regular Season	S	Kevin Huerter	ATL	1628989	92	1610612737	74	4.7	10.3	0.454	2.2
17	PerGame	pts	2019-20	Regular Season	S	Trae Young	ATL	1629027	4	1610612737	60	9.1	20.8	0.437	3.4
18	PerGame	pts	2020-21	Regular Season	S	Trae Young	ATL	1629027	14	1610612737	63	7.7	17.7	0.438	2.2
19	PerGame	pts	2021-22	Regular Season	S	Trae Young	ATL	1629027	4	1610612737	76	9.4	20.3	0.460	3.1
20	PerGame	pts	2022-23	Regular Season	S	Trae Young	ATL	1629027	10	1610612737	73	8.2	19.0	0.429	2.1
21	PerGame	pts	2019-20	Regular Season	S	Cam Reddish	ATL	1629629	128	1610612737	58	3.7	9.6	0.384	1.4

Showing 1 to 21 of 833 entries, 33 total columns

```
library(ggplot2)

pl <- ggplot(last_four_seasons_data, aes(pctFG, pts, color = slugSeason))

print(pl + geom_point(size = 4))
```



We can count that of the data points that passed averaging 30 points; 6 of the 11 are purple and only 1 is red and the other 2 seasons (2020-2021 and 2021-2022) split the other 4 dots 2 and 2.

We will also use the **last_four_seasons_truncated** data frame we created earlier in the course which is a subset ot the **last_four_seasons_combined** data frame.

PROJECT CODE NBASTATR.R* | web scraper project.R* | advance_stats_2019_2020 | ml project k means clustering.R

Filter

	pctFG	pts	pctFG3	pctFT	namePlayer	slugSeason
1	0.444	34.3	0.3548387	0.865	James Harden	2019-20
2	0.463	30.0	0.4019608	0.888	Damian Lillard	2019-20
3	0.437	29.6	0.3578947	0.860	Trae Young	2019-20
4	0.553	29.5	0.2978723	0.633	Giannis Antetokounmpo	2019-20
5	0.463	28.8	0.3146067	0.758	Luka Doncic	2019-20
6	0.489	26.6	0.3508772	0.919	Devin Booker	2019-20
7	0.503	26.1	0.3428571	0.846	Anthony Davis	2019-20
8	0.450	25.5	0.3827160	0.802	Zach LaVine	2019-20
9	0.493	25.3	0.3492063	0.693	LeBron James	2019-20
10	0.449	24.0	0.3623188	0.863	Donovan Mitchell	2019-20
11	0.463	23.8	0.3870968	0.851	Brandon Ingram	2019-20
12	0.450	23.4	0.4084507	0.812	Jayson Tatum	2019-20
13	0.453	22.9	0.3606557	0.792	Pascal Siakam	2019-20
14	0.451	22.2	0.3835616	0.757	CJ McCollum	2019-20
15	0.531	22.1	0.2000000	0.845	DeMar DeRozan	2019-20
16	0.497	20.9	0.4210526	0.916	Khris Middleton	2019-20
17	0.472	20.8	0.3846154	0.846	Collin Sexton	2019-20
18	0.415	20.6	0.3015873	0.778	Spencer Dinwiddie	2019-20
19	0.447	20.2	0.4109589	0.903	Bojan Bogdanovic	2019-20
20	0.455	19.9	0.2380952	0.834	Jimmy Butler	2019-20
21	0.528	19.9	0.3142857	0.817	Nikola Jokic	2019-20

Showing 1 to 21 of 833 entries, 6 total columns

Now we can develop our concise K-means algorithm. We will name it **nba.data.cluster** and since this algorithm is unsupervised meaning it does not take in labels we need to feed it the scoring statistics **(pctFG, pts, pctFG3, and pctFT)** with the player and season columns omitted [,1:4] and we want to use 4 clusters; one for each of the seasons we've been analyzing. The **nstart** part of the code represents the number of times the k-means clustering process will be run with different initial cluster centroids. The algorithm starts by randomly initializing K cluster centroids and assigns data points to their nearest centroid. However, since the initial centroids are chosen randomly, the resulting clustering solution may vary. By setting the **nstart** parameter to a value greater than 1, the k-means algorithm is repeated multiple times with different random initializations. In our case we chose 20 but the Cluster Mean results (as you can see below won't vary much if you choose a different number and support the hypothesis of there being very small differences between the seasons.

```
nba.data.cluster <- kmeans(last_four_seasons_truncated[,1:4],4,nstart = 20)

 print(nba.data.cluster)
```

```
K-means clustering with 4 clusters of sizes 292, 291, 67, 183

Cluster means:
     pctFG       pts    pctFG3     pctFT
1 0.4751712 11.208904 0.3329577 0.7783185
2 0.4738900  6.645017 0.3138394 0.7423677
3 0.4886418 26.722388 0.3598145 0.8208209
4 0.4705683 17.931148 0.3502943 0.8031858
```

```
Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [82] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[163] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1
[244] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[325] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
[406] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[487] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[568] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
[649] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[730] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[811] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Within cluster sum of squares by cluster:
[1] 735.2573 648.3232 539.4712 880.8503
 (between_SS / total_SS =  91.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
"iter"         "ifault"
```

Because we indeed have the labels to our data, we can compare against actual values. We create a table and pull in our **nba.data.cluster** with the **cluster** function along with our **last_four_seasons_truncate** with the slugSeason column

```
# checking against actual values

table(nba.data.cluster$cluster,last_four_seasons_truncated$slugSeason)

   2019-20 2020-21 2021-22 2022-23
 1      64      63      79      81
```

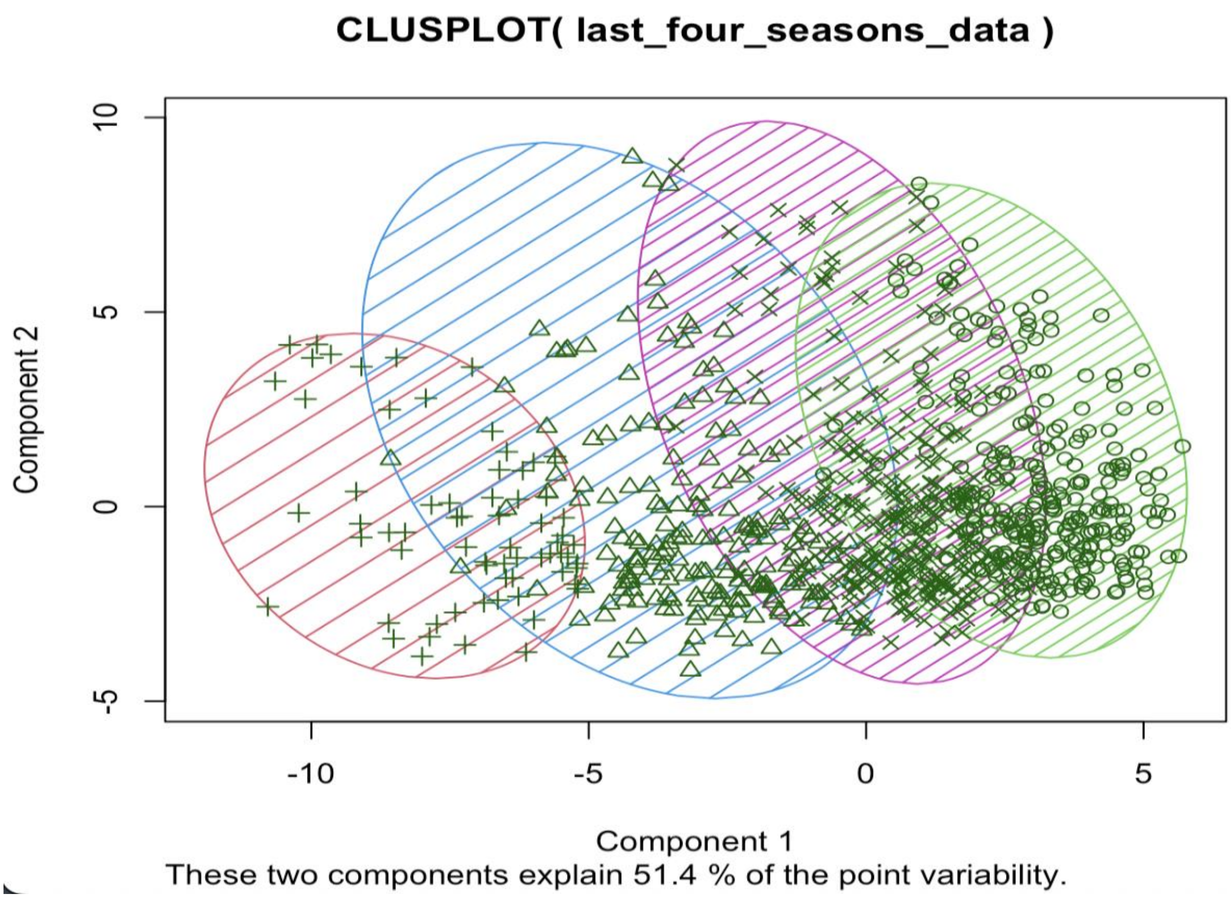
2	**44**	**28**	**56**	**55**
3	**13**	**17**	**15**	**22**
4	**68**	**63**	**78**	**87**

Each row represents the 4 metrics we passed into the algorithm and above you see the season labels. Again, we can clearly see that across each metric the numbers are comparable amongst the common ball manufacturers season, but the data shows a trend upwards in offensive production and efficiency after Wilson took over.

Lastly, we plot our **clusters**; make sure to call in the clusters library and set labels to 0 in the code.

```
# plotting above clusters

library(cluster)
clusplot(last_four_seasons_data,nba.data.cluster$cluster, color = T, shade = T, labels = 0, lines = 0)
```



Our K-means clustering was a success! And it looks like it supports the narrative we've been noticing since exploring and working with this data. On aggregate most of the points accumulated near the same areas meaning that the 4 seasons we're close in statistics but there was a noticeable trend in offensive production once Wilson took over.

The "+" represents the data points for the 2019-2020 season.
The "△" represents the data points for the 2020-2021 season.
The " x " represents the data points for the 2021-2022 season.
The "o" represents the data points for the 2022-2023 season.

The "+" is the weakest season and it shows here; this is for sure the 2019-2020 seasons. We can see the "△" and "x" mostly jumbled up in the center of the graph which were the 2 seasons with the closest data. The "o" points are scattered to the right which is the season with the highest offensive production of the 4.

K-means was able explain 51.4% of the point variability using Principal Component Analysis (PCA) which is a statistical technique used in conjunction with k-means clustering. It aims to reduce the dimensionality of the data by transforming it into a new set of uncorrelated variables called principal components. These components capture the most important patterns and variations in the data. By applying PCA before k-means clustering in R, we can focus on the most informative components, potentially improving the clustering results and making them more interpretable.

PCA is particularly useful when dealing with datasets containing many variables or when seeking a compressed representation of the data. It helps identify the most important patterns and relationships among variables, facilitates data visualization, and can be utilized as a preprocessing step for various machine learning algorithms. It is crucial to compare the results of k-means with domain knowledge and assess if the obtained clusters are meaningful and useful for the intended analysis or application.

i.e.: The amount of variance explained is related to the two principal components calculated to visualize your data. This has nothing to do with the type of clustering algorithm or the accuracy of the algorithm that we're using!

Results

In my analysis of NBA data, I found no substantial evidence to support the claim that the transition from Spalding to Wilson basketballs has had a noticeable effect on scoring. After carefully examining the relevant statistical measures, such as points per game and shooting percentages, across multiple seasons, I observed no significant variations that can be directly attributed to the change in basketball brands.

However, it is worth mentioning that there appears to be a slight upward trend in scoring during the 2022-2023 season. Although this trend could be a result of various factors unrelated to the basketball change, it may warrant further investigation to fully understand its origins. It is also worth mentioning 2 things first, this was a shortened offseason, so the players had less time off before the start of the first Wilson season; and that pace of play and points scored in a game are increasing which aligns with our data supports. “NBA teams have never been better at scoring points than they are right now. This statement has been uttered many times in the past decade as the league has upped the ante on 3-point attempts year after year after year.” The Athletic NBA Staff. (2023)

“So, what’s going on? To start with, teams as a whole are scoring more. The average N.B.A. team has scored 113.8 points a game this year, the highest total since 1970. Ten years ago, the average was 98.1. The pace of games has also sped up, with teams averaging nearly 100 possessions every 48 minutes over the past five seasons, which had not been done since the 1980s. More possession, more shots, more points for everyone.” Mather. (2023)

	2019-2020	**2020-2021**	**2021-2022**	**2022-2023**
Field Goal %	46.6%	47.9%	46.9%	48.4%
Free Throw %	77.4%	77.5%	77.5%	77.4%
3 Point %	33.9%	33.3%	33.2%	32.7%
Points	12.2	12.3	12.2	12.6
Offensive Rating	111.3	113.1	112.9	115.7
Defensive Rating	111.4	113.1	112.9	115.7
Net Rating	- 0.1596667	-0.0003333333	-0.0003333333	1.532339e-17

References & Sources

Chow, A. (2021, November 9). *NBA Players Comment on New Wilson Game Basketball*. Hypebeast. https://hypebeast.com/2021/11/nba-players-complain-wilson-game-ball-change-info

Kmeans: K-means Clustering. RDocumentation. (n.d.). https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kmeans

Mangubat, L. (2020, May 14). *NBA's New Deal With Wilson: Same Ball, Different Logo*. Spin.ph. https://www.spin.ph/life/guide/new-wilson-ball-for-the-nba-same-rock-different-logo-a1017-20200514

Mather, V. (2023, January 6). *As the N.B.A. gets "loose," 50-point games are surging*. The New York Times. https://www.nytimes.com/2023/01/06/sports/basketball/donovan-mitchell-50-point-games.html#:~:text=The%20average%20N.B.A.,been%20done%20since%20the%201980s.

Staff, T. A. N. (2023, January 6). *NBA's Scoring Explosion: What's Happening, Can it Continue And Will Anyone Score 82 (or more)?* The Athletic. https://theathletic.com/4063884/2023/01/06/nbas-scoring-explosion/

Sundaresan, M. (2021, November 9). *NBA Stars Are Missing 3-Pointers This Season. Is a New Ball Really to Blame?*. NPR. https://www.npr.org/2021/11/09/1052884749/nba-stars-are-missing-3-pointers-new-wilson-ball-blamed