DATA ANALYTICS

Boubacar Traoré - Décembre 2022

Impact of a player on team performance :

The case for Giannis Antetokounmpo being the best player in the NBA

Table of contents

Introduction **3**

Data and data sources **4**

Data cleaning and Exploratory data analysis **6**

Python cleaning process **6**

Data visualization **9**

Database **12**

ER Model **12**

Database selection **13**

SQL Queries **14**

Conclusion **16**

Introduction

Performances of the Milwaukee bucks and its superstar Giannis Antetokounmpo

The NBA (National Basketball Association) is a professional basketball league in North America and the biggest basketball league in the world. The league is composed of 30 teams (29 in the United States and 1 in Canada).

Unlike many team sports where each team has developed an identity of play, the style of play of NBA teams varies from year to year and mainly depends on the star player. It only takes one trade to make a champion team at the bottom of the standings the next year.

Passionate about basketball I am following the NBA and most particularly the Bucks, the team of Milwaukee city. Giannis Antetokounmpo is a Greek player who was drafted by the bucks in 2013, at the age of 18. He is now 28 and one of the best players in the NBA with two MVP (Most Valuable Trophies) in 2019 and 2020.

Having learnt the basic tools of data analysis I now want to do my own approach on analyzing data about the evolution of the Bucks

The goal of this project was to analyze the weight of Giannis Antetokounmpo performances on the results of the Milwaukee Bucks team by comparing the player individuals' statistics and the team statistics. I also want to show the evolution of the Milwaukee Bucks across the years team since Giannis started being in the team in 2013.

The plan of the project was to choose a data source that was able to provide all the game results and statistics of the Milwaukee bucks since 2013.

After export of the data into CSV files, imported the files into a new script of Python, to produce data cleaning (by managing with outliers, missing values, and features that were not relevant for the study) and visualizations, to perform a primary data analysis.

DATA AND DATA SOURCES

The data sources are from (https://www.kaggle.com/datasets/nathanlauga/nba-games?resource=download)

It is composed of 5 CSV files: games.csv, games_details.csv, players.csv, ranking.csv, teams.csv.
I have only used these three files: games.csv, games_details.csv.
I needed to merge these two tables to compare games statistics with players statistics

file	Games.csv	Games_details.csv
Content	Every NBA game from 2003 to 2022 with stats of the two teams	Every NBA game from 2003 to 2022 with stats for each player of the two teams
shape	25 796 rows, 21 columns	645 953 rows, 29 columns

Description of some not-explicit columns:

Games.csv

'PTS_home/away': Total points scored by the home/away team
'AST_ home/away': Total assists (pass before a shot made) by the home/away team
'REB_ home/away': Total rebound by the home/away team

'FG_PCT_ home/away': percentage (accuracy) of shot made by the home/away team
'FT_PCT_ home/away': percentage (accuracy) of free throws made by the home/away team
'FG3_PCT_ home/away': percentage (accuracy) of 3 points shots made by the home/away team

Games_details.csv

‘START_POSITION': role of the player when he started the game

'MIN': total minutes played by a player during a game

'FGA': (Field goal attempts) number of shots attempts by a player

'FGM': (Field goal made) number of shots made by a player

'FG_PCT':(Field goal percentage) = FGM / FGA

'OREB': offensive rebounds by a player

'DREB': defensive rebounds by a player

'REB': total rebounds by a player

'AST': total assists by a player

'STL': total steal by a player

'BLK': total block by a player

'TO': total turnover by a player

'PTS': total points by a player

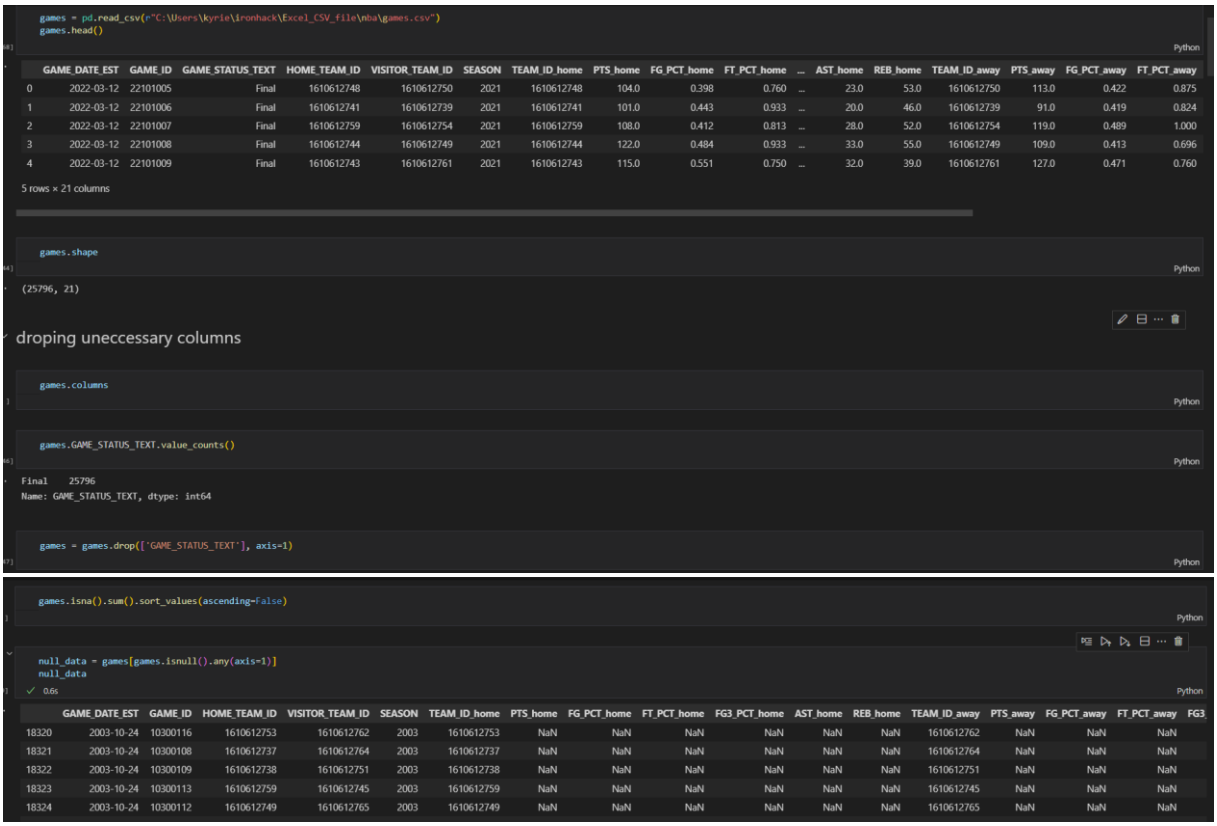
'PLUS_MINUS'**:** used to measure a player's impact on the game, represented by the difference between their team's total scoring versus their opponent's when the player is in the game.

Data cleaning and Exploratory data analysis

Python cleaning process (cleaning of the two files separately)

Games.csv

- Import of the libraries Pandas and NumPy
- Import files with pandas to read csv
- Listed the name of the columns to verifier
- Delete unnecessary columns (columns with one unique value and columns that I don't want to use)
- Look for missing values in the data frame: there only missing values for the 2003 season. We want to focus on the year where Giannis Antetokounmpo was playing (2013-2022)
- Look for outliers with boxplots visualization and drop the outliers
- Create a new data frame with the data from 2013 to 2022 only
- Export the data frame to csv ("cleangames.csv") to perform visualization



```
games = pd.read_csv(r"C:\Users\kyrie\ironhack\Excel_CSV_file\nba\games.csv")
games.head()
```

Python

	GAME_DATE_EST	GAME_ID	GAME_STATUS_TEXT	HOME_TEAM_ID	VISITOR_TEAM_ID	SEASON	TEAM_ID_home	PTS_home	FG_PCT_home	FT_PCT_home	...	AST_home	REB_home	TEAM_ID_away	PTS_away	FG_PCT_away	FT_PCT_away
0	2022-03-12	22101005	Final	1610612748	1610612750	2021	1610612748	104.0	0.398	0.760	...	23.0	53.0	1610612750	113.0	0.422	0.875
1	2022-03-12	22101006	Final	1610612741	1610612739	2021	1610612741	101.0	0.443	0.933	...	20.0	46.0	1610612739	91.0	0.419	0.824
2	2022-03-12	22101007	Final	1610612759	1610612754	2021	1610612759	108.0	0.412	0.813	...	28.0	52.0	1610612754	119.0	0.489	1.000
3	2022-03-12	22101008	Final	1610612744	1610612749	2021	1610612744	122.0	0.484	0.933	...	33.0	55.0	1610612749	109.0	0.413	0.696
4	2022-03-12	22101009	Final	1610612743	1610612761	2021	1610612743	115.0	0.551	0.750	...	32.0	39.0	1610612761	127.0	0.471	0.760

5 rows × 21 columns

```
games.shape
```

Python

```
(25796, 21)
```

droping uneccessary columns

```
games.columns
```

Python

```
games.GAME_STATUS_TEXT.value_counts()
```

Python

```
Final    25796
Name: GAME_STATUS_TEXT, dtype: int64
```

```
games = games.drop(['GAME_STATUS_TEXT'], axis=1)
```

Python

```
games.isna().sum().sort_values(ascending=False)
```

Python

```
null_data = games[games.isnull().any(axis=1)]
null_data
```

0.6s Python

	GAME_DATE_EST	GAME_ID	HOME_TEAM_ID	VISITOR_TEAM_ID	SEASON	TEAM_ID_home	PTS_home	FG_PCT_home	FT_PCT_home	FG3_PCT_home	AST_home	REB_home	TEAM_ID_away	PTS_away	FG_PCT_away	FT_PCT_away	FG3
18320	2003-10-24	10300116	1610612753	1610612762	2003	1610612753	NaN	NaN	NaN	NaN	NaN	NaN	1610612762	NaN	NaN	NaN	
18321	2003-10-24	10300108	1610612737	1610612764	2003	1610612737	NaN	NaN	NaN	NaN	NaN	NaN	1610612764	NaN	NaN	NaN	
18322	2003-10-24	10300109	1610612738	1610612751	2003	1610612738	NaN	NaN	NaN	NaN	NaN	NaN	1610612751	NaN	NaN	NaN	
18323	2003-10-24	10300113	1610612759	1610612745	2003	1610612759	NaN	NaN	NaN	NaN	NaN	NaN	1610612745	NaN	NaN	NaN	
18324	2003-10-24	10300112	1610612749	1610612765	2003	1610612749	NaN	NaN	NaN	NaN	NaN	NaN	1610612765	NaN	NaN	NaN	

```
games2 = games.loc[games.SEASON > 2012]
games2
```

✓ 0.4s Python

	GAME_DATE_EST	GAME_ID	HOME_TEAM_ID	VISITOR_TEAM_ID	SEASON	TEAM_ID_home	PTS_home	FG_PCT_home	FT_PCT_home	FG3_PCT_home	AST_home	REB_home	TEAM_ID_away	PTS_away	FG_PCT_away	FT_PCT_away	FG3
0	2022-03-12	22101005	1610612748	1610612750	2021	1610612748	104.0	0.398	0.760	0.333	23.0	53.0	1610612750	113.0	0.422	0.875	
1	2022-03-12	22101006	1610612741	1610612739	2021	1610612741	101.0	0.443	0.933	0.429	20.0	46.0	1610612739	91.0	0.419	0.824	
2	2022-03-12	22101007	1610612759	1610612754	2021	1610612759	108.0	0.412	0.813	0.324	28.0	52.0	1610612754	119.0	0.489	1.000	
3	2022-03-12	22101008	1610612744	1610612749	2021	1610612744	122.0	0.484	0.933	0.400	33.0	55.0	1610612749	109.0	0.413	0.696	
4	2022-03-12	22101009	1610612743	1610612761	2021	1610612743	115.0	0.551	0.750	0.407	32.0	39.0	1610612761	127.0	0.471	0.760	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
25791	2014-10-06	11400007	1610612737	1610612740	2014	1610612737	93.0	0.419	0.821	0.421	24.0	50.0	1610612740	87.0	0.366	0.643	
25792	2014-10-06	11400004	1610612741	1610612764	2014	1610612741	81.0	0.338	0.719	0.381	18.0	40.0	1610612764	85.0	0.411	0.636	
25793	2014-10-06	11400005	1610612747	1610612743	2014	1610612747	98.0	0.448	0.682	0.500	29.0	45.0	1610612743	95.0	0.387	0.659	
25794	2014-10-05	11400002	1610612761	1610612758	2014	1610612761	99.0	0.440	0.771	0.333	21.0	30.0	1610612758	94.0	0.469	0.725	
25795	2014-10-04	11400001	1610612748	1610612740	2014	1610612748	86.0	0.431	0.679	0.333	18.0	42.0	1610612740	98.0	0.462	0.706	

11992 rows × 20 columns

```
games2.to_csv(r"C:\Users\kyrie\ironhack-Final-Project\final_project\cleangames.csv")
```

✓ 0.1s Python

Gamesdetails.csv

- Import of the libraries Pandas and NumPy
- Import files with pandas to read csv
- Listed the name of the columns to verifier
- Look for the columns type
- Delete unnecessary columns
- Look for and manage with the missing values
- Drop the rows with too many missing values
- Convert the ‘MIN’ column from object type to float type
- Had to replace Nan values by ‘F’ the “START_POSITION” column because Giannis Antetokounmpo role is Forward
- Create a new data frame (called giannis) which focus on the game where the Milwaukee Bucks and Giannis Antetokounmpo were playing
- Export the giannis data frame to csv (“cleangames.csv”) to perform visualization

```
players = pd.read_csv(r"C:\Users\kyrie\ironhack\Excel_CSV_file\nba\games_details.csv")
players.head(20)
```

Python

```
C:\Users\kyrie\AppData\Local\Temp\ipykernel_13904\47981702.py:1: DtypeWarning: Columns (6) have mixed types. Specify dtype option on import or set low_memory=False.
  players = pd.read_csv(r"C:\Users\kyrie\ironhack\Excel_CSV_file\nba\games_details.csv")
```

	GAME_ID	TEAM_ID	TEAM_ABBREVIATION	TEAM_CITY	PLAYER_ID	PLAYER_NAME	NICKNAME	START_POSITION	COMMENT	MIN	...	OREB	DREB	REB	AST	STL	BLK	TO	PF	PTS	PLUS_MINUS
0	22101005	1610612750	MIN	Minnesota	1630162	Anthony Edwards	Anthony	F	NaN	36:22	...	0.0	8.0	8.0	5.0	3.0	1.0	1.0	1.0	15.0	5.0
1	22101005	1610612750	MIN	Minnesota	1630183	Jaden McDaniels	Jaden	F	NaN	23:54	...	2.0	4.0	6.0	0.0	0.0	2.0	2.0	6.0	14.0	10.0
2	22101005	1610612750	MIN	Minnesota	1626157	Karl-Anthony Towns	Karl-Anthony	C	NaN	25:17	...	1.0	9.0	10.0	0.0	0.0	0.0	3.0	4.0	15.0	14.0
3	22101005	1610612750	MIN	Minnesota	1627736	Malik Beasley	Malik	G	NaN	30:52	...	0.0	3.0	3.0	1.0	1.0	0.0	1.0	4.0	12.0	20.0
4	22101005	1610612750	MIN	Minnesota	1626156	D'Angelo Russell	D'Angelo	G	NaN	33:46	...	0.0	6.0	6.0	9.0	1.0	0.0	5.0	0.0	14.0	17.0
5	22101005	1610612750	MIN	Minnesota	1629675	Naz Reid	Naz	NaN	NaN	23:56	...	3.0	7.0	10.0	1.0	3.0	2.0	1.0	1.0	11.0	-7.0
6	22101005	1610612750	MIN	Minnesota	1629162	Jordan McLaughlin	Jordan	NaN	NaN	21:00	...	0.0	1.0	1.0	3.0	3.0	0.0	0.0	1.0	5.0	-10.0
7	22101005	1610612750	MIN	Minnesota	1629669	Jaylen Nowell	Jaylen	NaN	NaN	21:35	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	16.0	-5.0
8	22101005	1610612750	MIN	Minnesota	1627752	Taurean Prince	Taurean	NaN	NaN	22:53	...	0.0	2.0	2.0	1.0	1.0	0.0	1.0	2.0	11.0	1.0
9	22101005	1610612750	MIN	Minnesota	1629006	Josh Okogie	Josh	NaN	NaN	0:25	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	22101005	1610612750	MIN	Minnesota	1630195	Leandro Bolmaro	Leandro	NaN	DNP - Coach's Decision	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
11	22101005	1610612750	MIN	Minnesota	1630233	Nathan Knight	Nathan	NaN	DNP - Coach's Decision	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12	22101005	1610612750	MIN	Minnesota	1627774	Jake Layman	Jake	NaN	DNP - Coach's Decision	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
13	22101005	1610612748	MIA	Miami	1629130	Duncan Robinson	Duncan	F	NaN	16:31	...	0.0	0.0	0.0	4.0	0.0	0.0	0.0	3.0	3.0	-21.0
14	22101005	1610612748	MIA	Miami	200782	P.J. Tucker	P.J.	F	NaN	23:17	...	4.0	3.0	7.0	0.0	0.0	0.0	3.0	4.0	6.0	-25.0
15	22101005	1610612748	MIA	Miami	1628389	Bam Adebayo	Bam	C	NaN	33:28	...	2.0	10.0	12.0	4.0	3.0	0.0	4.0	2.0	19.0	0.0
16	22101005	1610612748	MIA	Miami	1629216	Gabe Vincent	Gabe	G	NaN	25:23	...	0.0	1.0	1.0	3.0	2.0	0.0	1.0	2.0	2.0	-22.0
17	22101005	1610612748	MIA	Miami	200768	Kyle Lowry	Kyle	G	NaN	37:24	...	2.0	5.0	7.0	7.0	0.0	0.0	3.0	2.0	14.0	-7.0
18	22101005	1610612748	MIA	Miami	1629639	Tyler Herro	Tyler	NaN	NaN	36:30	...	0.0	7.0	7.0	2.0	2.0	0.0	1.0	1.0	30.0	11.0
19	22101005	1610612748	MIA	Miami	1629622	Max Strus	Max	NaN	NaN	31:18	...	1.0	6.0	7.0	2.0	0.0	0.0	1.0	5.0	19.0	14.0

```
milwaukee = players.loc[players.TEAM_CITY == "Milwaukee"]
```

Python

```
giannis = milwaukee.loc[milwaukee.PLAYER_NAME == "Giannis Antetokounmpo"]
giannis
```

Python

	GAME_ID	TEAM_ID	TEAM_ABBREVIATION	TEAM_CITY	PLAYER_ID	PLAYER_NAME	NICKNAME	START_POSITION	COMMENT	MIN	...	OREB	DREB	REB	AST	STL	BLK	TO	PF	PTS	PLUS_MINUS
72	22101008	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F	NaN	34:01	...	1.0	7.0	8.0	3.0	0.0	1.0	1.0	2.0	31.0	-26.0
555	22100984	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F	NaN	37:03	...	3.0	9.0	12.0	5.0	1.0	0.0	2.0	4.0	43.0	3.0
859	22100979	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F	NaN	27:46	...	1.0	6.0	7.0	7.0	3.0	1.0	1.0	3.0	39.0	18.0
1144	22100961	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F	NaN	31:13	...	3.0	10.0	13.0	6.0	1.0	3.0	3.0	6.0	19.0	-8.0
1556	22100949	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F	NaN	38:48	...	4.0	12.0	16.0	5.0	2.0	1.0	4.0	4.0	34.0	15.0

```
minute = giannis['MIN']
minute = minute.str.replace(':', '.')
minute = minute.astype('float')
minute = round(minute)
```

Python

```
giannis.START_POSITION.value_counts()
```

Python

```
F    648
G     28
C      1
Name: START_POSITION, dtype: int64
```

```
plusminusmean = giannis.PLUS_MINUS.mean()
```

Python

```
giannis['START_POSITION']= giannis['START_POSITION'].replace(np.nan,'F')
giannis['PLUS_MINUS']= giannis['PLUS_MINUS'].replace(np.nan,plusminusmean)
```

Python

```
giannis.to_csv(r"C:\Users\kyrie\Ironhack-Final-Project\final_project\giannis.csv")
```

Python

Merge the two data frame and perform visualization

```
df = pd.merge(games, giannis, how='inner', on = 'GAME_ID')
df
```

Python

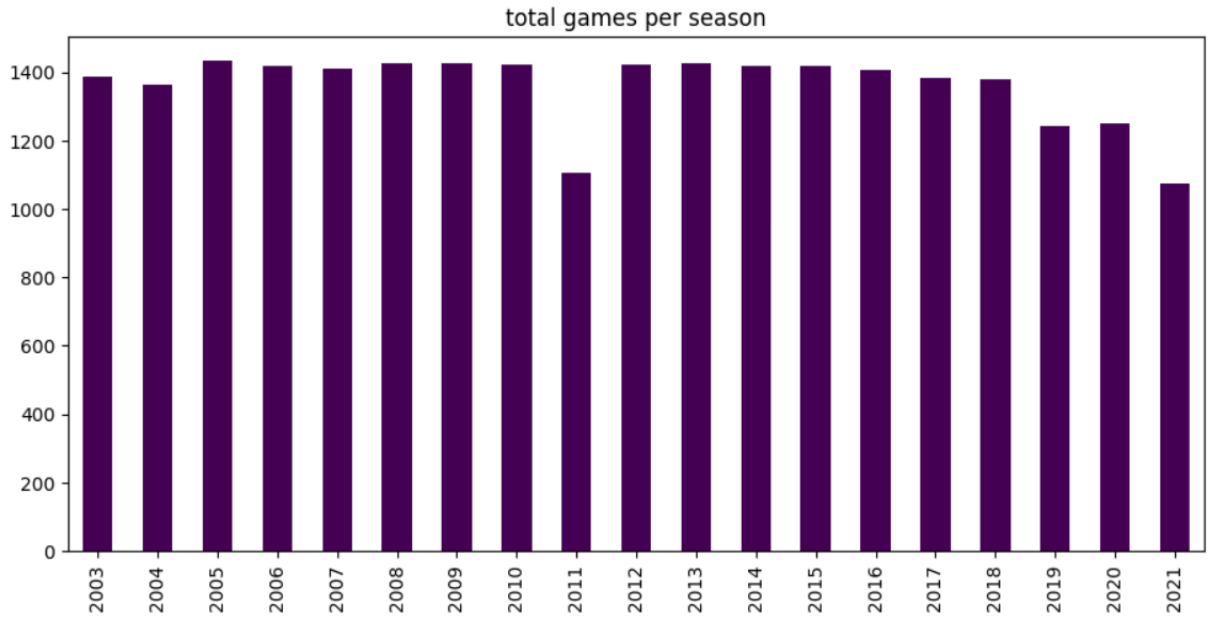
```
df.shape
```

✓ 0.3s Python

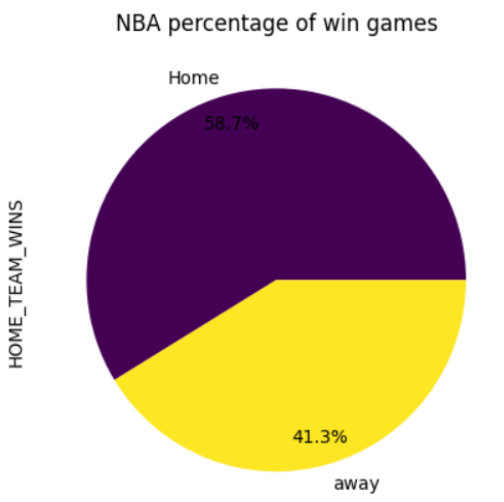
```
(755, 41)
```

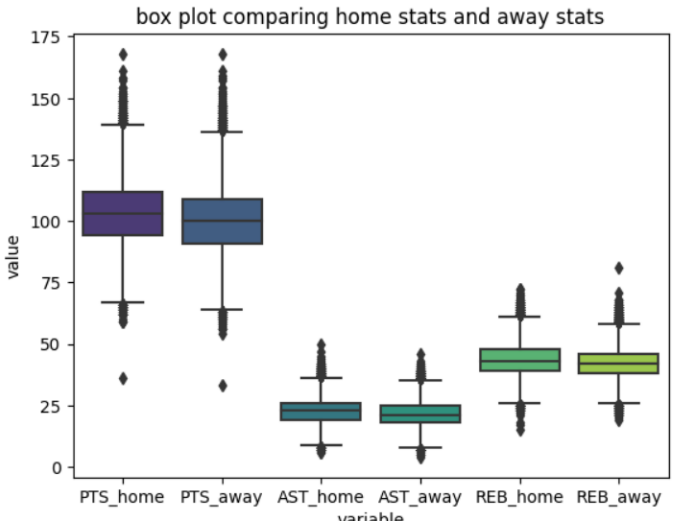
Data visualization

Visualization about the NBA League:



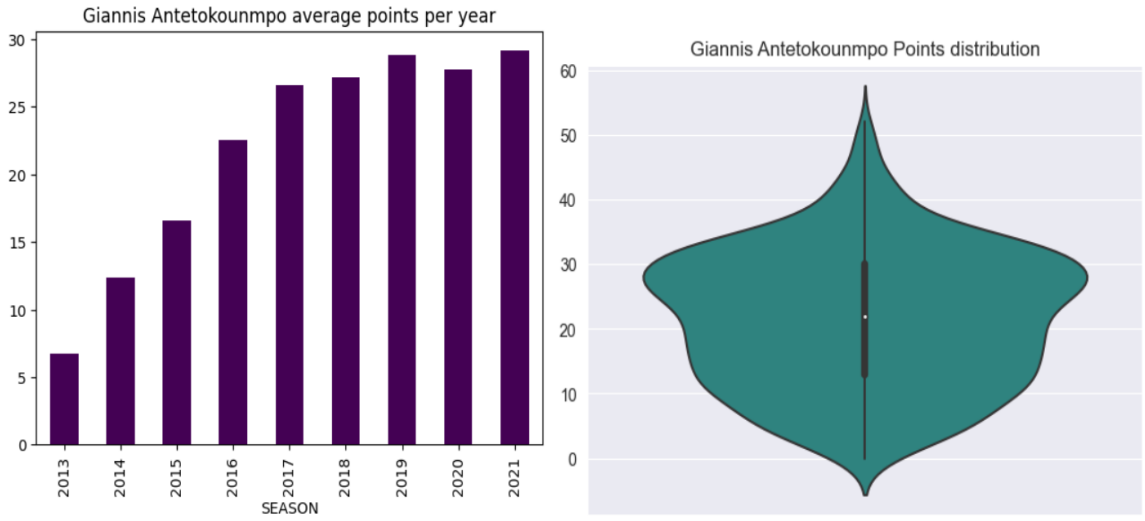
The previous lockout in 1998–99 had shortened the season to 50 games. During the lockout, teams could not trade, sign, or contact players.





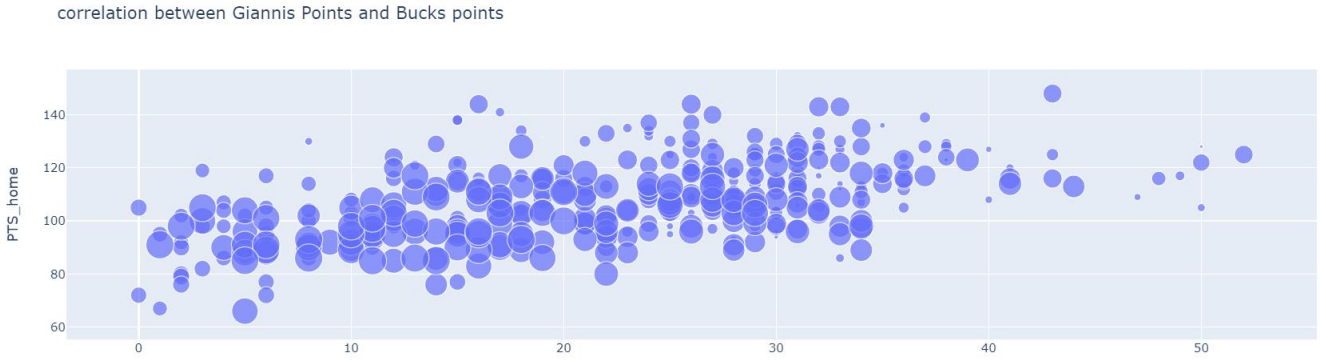
We can see that when a team plays Home, its win rate is higher. And the main stats (points, rebound and assists) in the boxplot follow the tendency by being a little higher for home team.

Visualization about Giannis performances:



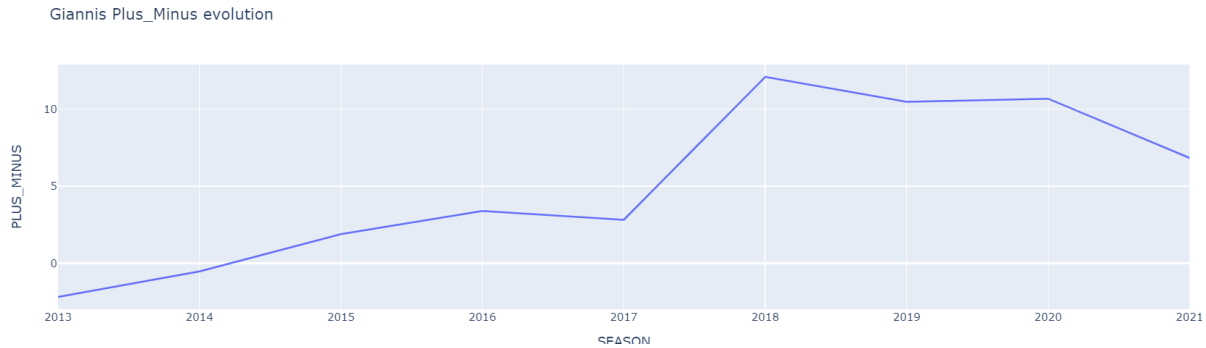
Giannis was getting better and improving his game year by year until he become the best player in the league in 2019 and 2020.

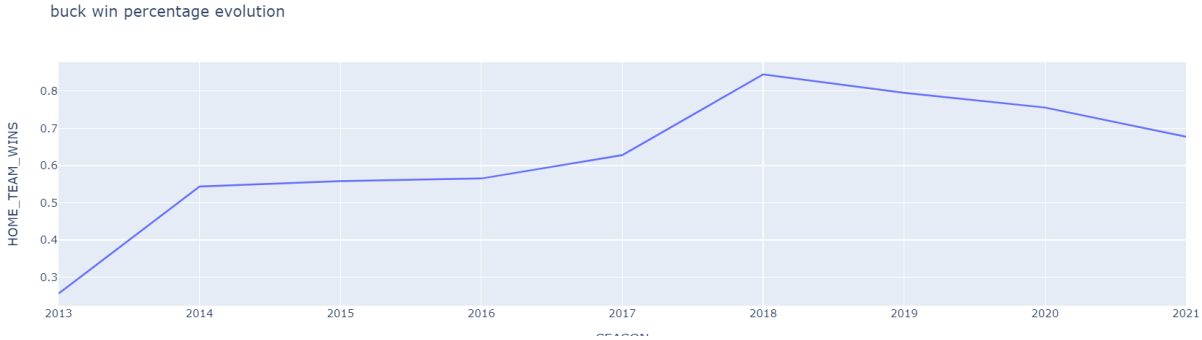
Giannis weight on the bucks' team:

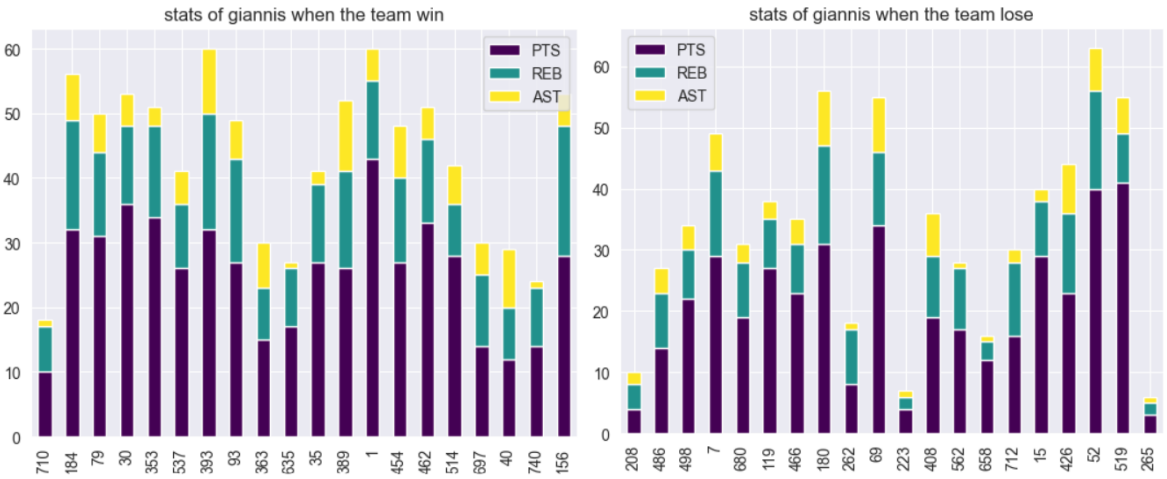


In the two graphs below, we compare Giannis' evolution as a player with the win rate of the Milwaukee bucks.

These two graphs almost have the same behavior. So, it shows that Giannis has a huge impact on his team. He is a 'game changer'


Giannis Plus_Minus evolution
PLUS_MINUS
10
5
0
2013
2014
2015
2016
2017
2018
2019
2020
2021
SEASON


buck win percentage evolution
HOME_TEAM_WINS
0.8
0.7
0.6
0.5
0.4
0.3
2013
2014
2015
2016
2017
2018
2019
2020
2021


stats of giannis when the team win
PTS
REB
AST
60
50
40
30
20
10
0
710
184
79
30
353
537
393
93
363
635
35
389
1
454
462
514
697
40
740
156

stats of giannis when the team lose
PTS
REB
AST
60
50
40
30
20
10
0
208
486
498
7
680
119
466
180
262
69
223
408
562
658
712
15
426
52
519
265

Database selection

There are several reasons why one might choose to use a SQL database rather than a NoSQL database:

1. SQL databases are designed to store and manage structured data, that is, data organized into well-defined tables and columns. This makes SQL databases well suited for applications that require complex queries and transactions, such as online banking or e-commerce.

2. SQL databases are based on a relational model, which allows them to easily link data from different tables using primary and foreign keys. This makes it easy to query and update data in an SQL database, as well as enforce data integrity and consistency.

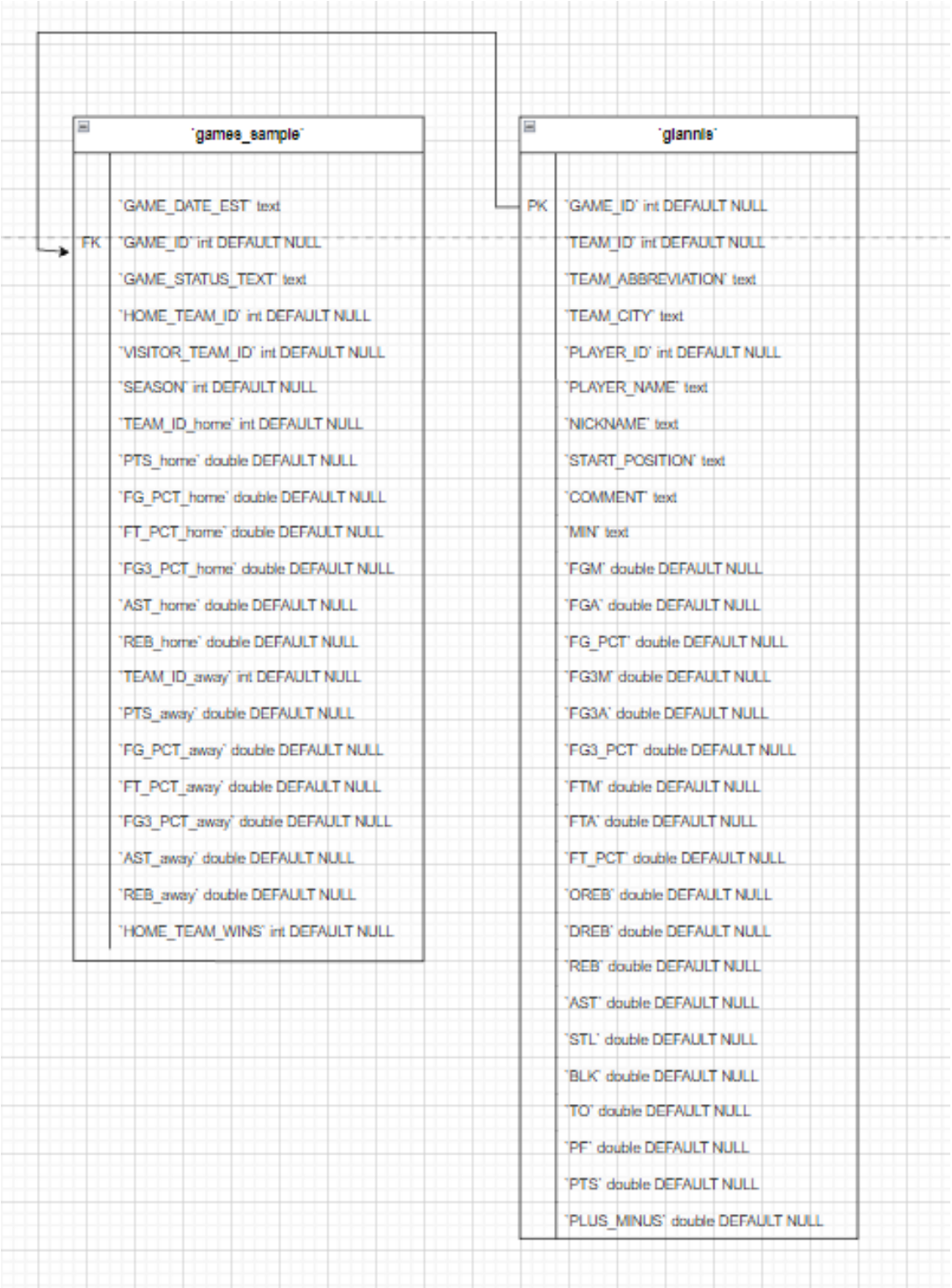
3. SQL databases are highly scalable, which means they can handle large amounts of data and concurrent users without sacrificing performance. This makes SQL databases a good choice for high traffic and high data volume applications.

4. SQL databases are supported by a wide range of tools and technologies, including various programming languages, frameworks, and libraries. This makes it easy to integrate SQL databases into existing applications and to develop new applications using SQL databases.

In summary, SQL databases are well suited for applications that require complex queries, transactions, scalability, and data integrity, and are supported by a wide range of tools and technologies.

SQL	NOSQL
Relation Database management system	Distributed Database management system
Vertically scalable	Horizontally scalable
Fixed or predefined Schema	Dynamic Schema
Not suitable for hierarchical data storage	Best suitable for hierarchical data storage
Can be used for complex queries	Not good for complex queries

Entities. ER models



SQL Queries

According to the previous part, I decided to use SQL

I had some problems while importing my data in MySQL because the file was really big, so I used sample of my two data frame and export them as csv:

```python
players_sample = players.head(1000)
players_sample.to_csv(r"C:\Users\kyrie\ironhack\final_project\players_sample.csv")
```

Python

```python
games_sample = games.head(1000)
games_sample.to_csv(r"C:\Users\kyrie\ironhack\final_project\games_sample.csv")
```

Python

- Create database and locate the Milwaukee team id in the table ‘players_sample’.

```
Limit to 100 rows
1  create database nba;
2  use nba;
3
4  SELECT TEAM_ID FROM players_sample
5  WHERE TEAM_CITY = "milwaukee";
6
7  # milwaukee team ID is 1610612749
```

- Display the game where Giannis Antetokounmpo has played
- Display the average points, assists, and rebound of Giannis Antetokounmpo
- Display the players who score more than 10 points, 10 assists, and 10 rebounds. In basketball its “called a triple double”

```
9  SELECT * FROM games_sample
10 WHERE TEAM_ID_home = 1610612749 OR TEAM_ID_away = 1610612749;
11
12 SELECT * FROM players_sample AS players
13 WHERE PLAYER_NAME = "Giannis Antetokounmpo";
14 SELECT avg(PTS), avg(REB), avg(AST) FROM players_sample AS players
15 WHERE PLAYER_NAME = "Giannis Antetokounmpo";
16
17 SELECT * FROM players_sample
18 WHERE PTS >= 10 AND AST >=10 AND REB >= 10;
```

- create a table which focus on Giannis Antetokounmpo statistics
- merge this table with the game-sample table to compare Giannis stats with the bucks team stats

```
20 •  create table giannis
21    select * from players_sample where PLAYER_NAME = "Giannis Antetokounmpo" ;
22
23
24 •  SELECT *
25    FROM games_sample gs
26    LEFT JOIN players_sample pl
27    using (GAME_ID)
28    having pl.PLAYER_NAME = "Giannis Antetokounmpo";
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

lumn	TEAM_ID	TEAM_ABBREVIATION	TEAM_CITY	PLAYER_ID	PLAYER_NAME	NICKNAME	START_POSITION	COMMENT	MIN	FGM	FGA	FG_PCT	FG3M	FG3A	FG3_PCT
	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F		34:01	9	17	0.529	1	4	0.25
	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F		37:03	15	22	0.682	1	5	0.2
	1610612749	MIL	Milwaukee	203507	Giannis Antetokounmpo	Giannis	F		27:46	13	19	0.684	4	4	1

Conclusion

The data I collected for this project was quite comprehensive. Indeed, the NBA datasets are fed very seriously and regularly by the NBA and by the fan's associations. For my project I used one dataset about the history of all the NBA games since 2003 and my second dataset was about the players statistics for all these games.

I cleaned the data from both files step by step and one dataset after the other. The first step was to select the columns I wanted to deal with and to drop the not relevant ones. Then I had to manage with some missing values and to drop some outliers. In the end I merge my two data frames to do some charts who compare Giannis Antetokounmpo with the Milwaukee Bucks team.

The visualization was about to show some characteristics of an NBA season and then I tried to show the correlation with Giannis Antetokounmpo statistics in game with the performances of his team. Some charts are clear by showing graphs with a linear behavior.

Finally, we can conclude that Giannis Antetokounmpo had a huge impact on the Milwaukee Bucks dominations this last few years as well as the Milwaukee team had an impact on the Giannis Antetokounmpo