

# NBA Statisical and Individual Player Analysis 2004, 2014, and 2023

## Authors: Spencer Gardner and Tyler Braham

### Summary

1. What in common do successful players have? What common traits do they have in common? (Messy Data)
  - To answer this question we used our web scraped data for the top 5 players in each year, 2004, 2014, and 2023. from our analysis, we compared a variety of factors and saw that Forward is the most common position among the Top 5 players. The majority of these players are also between the range of 200-249 lbs. The height varies, and with more data we speculate that this would also change. From this data, the most common heights were 6ft 7 and 6ft 8. The mean points per game for a top 5 player is 28.1, the mean assists per game for a top 5 player is 4.9, and rebounds is 7.6. From these analyses, this tells us that the top players do not just have a high number of points per game, they also have well rounded stats in other areas of the game such as assists and rebounds. This means the top 5 players from each year are not just scoring the points, but also affect the game in all areas, which makes them an overall successful player.
2. Have shot locations changed over time within the top players? (2004 - 2014 - 2023) (Multiple Datasets)
  - To answer this question, we had to manipulate our data sets so that we could see the top players shot locations out of all the years. we merged our data sets that had the player's names and the ones that had their shot locations over the year. We used a heatmap to visualize our data, which we referenced a github site for help. Because our data was displayed using a graph we could only identify the results using qualitative analysis. From this, we can conclude that over time, across 2004-2014-2023, it's more common to see shots that are farther from the paint (close to the hoop). If we look at the graphs side by side, we can see that there is more density behind the 3 point line as the years increase. the majority of shots are still certainly in the paint/under the hoop from layups, dunks, and other paint-location shots. The difference behind the 3 point line is drastic when going from 2014 to 2023. This is not an uncommon take as many have addressed that the game of basketball has changed recently with stars like Stephen Curry and other players who dominate from behind the 3 point line.
3. Has the top 50 player stats changed over time? (Messy Data)
  - To answer this question, we wanted to compare various statistics from the top 50 players in their respective seasons such as 3 point percentages, 2 point percentages, points per game, assists, and rebounds. To asnwer our question, for the 2003-2004 season the mean 3P% was 31.3, for the 2013 - 2014 season it was 32.616, and for 2022-2023 it was 35.482. This indicates that the average three point percentage is increasing slightly over time. This means that the game is evolving and players are becoming more and more accurate in their 3 point shooting percentages. We performed the same analysis but for Field Goal shooting as well and we saw that the average percentage for 2003-2004 is 45.148, for 2013-2014 it is 45.922, and for 2022-2023 it is 47.71. For the 2003-2004 season the mean points per game was 18.708, for 2013-2014 it was 19.708, and for 2022-2023 it was 23.318. We see a significant jump in average points per game from 2004 to 2023 showing that on average, players are scoring more points. From these points, analysis indicates that players are on average making more shots and being more accurate shooters over time. For assists per game, we also saw an increase in assists per game across the seasons, and for rebounds, we saw a decrease over time.
4. Who performs the best per quarter out of the top players? (2004-2014-2023)?
  - We used the same function that we used to answer our question about changing shot locations since we adapted it to be able to plot a scatterplot type of figure as well. We plotted our results in a subplot figure that shows each year and each quarter within that year among the top players data set. with this we can directly see the differences in how the best performance within each quarter differs through the year and quarters. We notice that there are more attempts in longer ranges as the years increase. Of course this depends on the player but the majority of players who performed the best, take these longer range shots and make them more often than others. We do see that in recent years there is a good amount of balance between these types of shots among players.

### Motivation and background

- Apart from both enjoying watching NBA games, we wanted to get deeper insights and analyses to the game through data analysis. The NBA from the early 2000s to the present day has evolved and we wanted to find what has stayed similar and what has changed over time. We also want to apply our knowledge and data analysis skills from CSE 163 and apply it to a topic we are interested in which is basketball. We also want to further analyze performance stats to see what players share in which they are successful in the game. Also, we want to see, if anything, has changed over time. Finding and knowing this analysis will give us further insight into the game and could also be used to better predict successful players, future stats and what players will be successful.

### Data Setting

- Webscraping data from the top 50 players and various stats such as player name, rebounds, points per game, steals, assits, etc from the 2003-2004 season, 2013-2014, and 2022-2023 regular seasons. The website we scraped data from is from the NBA directly and can be found here: <https://www.nba.com/stats>

- Using a shot data file from [https://github.com/DomSamangy/NBA\\_Shots\\_04\\_23](https://github.com/DomSamangy/NBA_Shots_04_23) that contains individual shot data for all players from from 2003-2004 season, 2013-2014, and 2022-2023 seasons. This was helpful to us beacuse it contianed the locations of where the shots were taken so we could make a map of where the shots were taken and see thier changes over time.
- Finding the Top 5 players from from 2003-2004 season, 2013-2014, and 2022-2023 seasons, and getting webscraping for their stats to find varaibales and stats they have in common. to be successful. We found collected differetn stats from before such as draft pick, height, weight, and position. Data was scraped from the NBA Stats which can be found here: <https://www.nba.com/stats>
- For the Years chosen for analysis: the shot data file, the earliest year they had was 2004, therefore we wanted to increment by around 10 years, so we chose the years 2004, 2014, and 2023. We would have chosen the curretn 2024 data but the season is not over yet at the time of this data analysis therefore we chose the most recent completed year. To keep the years consistent, all our scraping was done from the same years to keep the data consistent.
- Link to all our data that we put into a google drive folder:  
<https://drive.google.com/drive/u/1/folders/1zzkVvCYBG9S9UevzfBML9FuFXMXUnWyU>

Methods

- To achieve our answers for our second and fourth research question we had to start out with trying to plot our results. Initially we were just going to use a picture of a basketball court for our background of our shot locations. We had trouble getting the exact location and scaling right with this so we opted for drawing our background using matplotlib's patches and shapes. with reference to <https://github.com/bradleyfay/py-Goldsberry/blob/master/docs/Visualizing%20NBA%20Shots%20with%20py-Goldsberry.ipynb> for help of how to use this package correctly, we utilized <https://official.nba.com/rule-no-1-court-dimensions-equipment/> for the picture and dimensions of our court bakcground. We implemented this into a function so we could draw our court to showcase our results as accurately as possible.
1. For our first research question, we wanted to find similarities between the top 5 players in 2004, 2014, and 2023 to determine what makes a successful player in the NBA. To do this, we web scraped data from the NBA Stats website for each of the top 5 players in each respective year. We scraped details about the player such as position, height, weight, draft pick, team, PPG, APG, RPG, and country. We then wanted to make distributions about this data and compare and contrast our visualizations. We decided to compare the points per game, position, rebounds, assists, player height, and weight. There are other factors such as draft pick and country we decided not to compare because we did not think this would tell us much. All the top 5 players were round 1 picks, meaning they were a top prospect before joining the NBA. Also, for the country, there were more players from the USA in the earlier years, not until 2023 that it was more diverse. Therefore, we thought that this would've been a biased comparison because it was not until later that the NBA became more diverse and players from around the world joined. We made visualizations using plotly, as one of the challenge goals for the new library to show commonalities and make conclusions from the data.
  2. To view our results for the second question, we implemented a shot chart function that can plot all of the locations of shots throughout the years 2004, 2014, and 2023 out of only the top 50 players. we gave it a lot of personable parameters that allows us to show exactly what we want such as the specific player, year, date of game, quarter, and whether or not it was a scatter plot or a heatmap. we filtered the data that would be passed into this function through these parameters and then graphed using seaborn's jointplot to plot a kde heatmap of our data. figuring out how to use this jointplot was difficult and ended up using a reference from the same source that we used for the drawing court function. It is still very finicky and the output plot is not very well integrated but it seems this is as far as we could get. we plotted this heatmap onto our drawn court and that gives us our results.
  3. For our third research question, we wanted to see how the statistics from the top 50 players in 2004, 2014, and 2023 changed over time. For this, we wanted to use some sort of distribution because it would show various analyses of the data and the team. We decided to use a box plot for each respective season because it provides a visual distribution of the data as well as make an interactive plot that shows various statistics about the distribution. This includes a lot of looking at plotlys documentation for creating the plots. We wanted to compare a variety of stats such as 3 point percentage, 2 point and 3 point combined percentage(Field Goal), overall points per game, assists per game, and rebounds. We will create plots that show the distributions for the specific statistic and it will compare the distribution for each season that we will use to then make conclusions about the data.
  4. For our fourth question we utilized the same function as question 2 since we had made it so we could answer both just with different parameters. using the scatterplot version of our function, we were able to plot them on a subplot that shows every quarter from every year (a 3x4 subplot). this gives us an easy way to compare between each quarter's best performing player for every year. to find the best player we initially used our data set that included all the players and all their shots. this proved difficult because the way that we calculated the best performing player was the ratio of how many points they scored total for that specific quarter to the total amount of that specific quarters they scored in. this would give us skewed results. for example 'Gerald Wallace' would come up as the best performing player afer scoring only 8 points total but within one first quarter. We felt this was not accurate data so we switched our data set to the data that comprised of all the same data entries but only within the top 50 players of that year. then we also chose to gather the heatmaps for all of the players so we could compare where they took the shots and how often they did.

# Results

## Imports

```
In [1]: import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# This is imports for our new library challenge goal plotly for some of our visualizations.
!pip install plotly
import plotly.express as px
import plotly.graph_objects as go
#drawing our court
from matplotlib.patches import Circle, Rectangle, Arc
```

Requirement already satisfied: plotly in /opt/conda/lib/python3.10/site-packages (5.19.0)  
Requirement already satisfied: tenacity>=6.2.0 in /opt/conda/lib/python3.10/site-packages (from plotly) (8.2.3)  
Requirement already satisfied: packaging in /opt/conda/lib/python3.10/site-packages (from plotly) (23.2)

## Reading in the data using pandas

- We had to manipulate our initial data sets so that we could merge them easily and gather the data that we needed.

```
In [10]: shot_locations_2023 = pd.read_csv(
        'NBA_2023_Shots.csv').rename(columns = {'SEASON_1' : 'year'}).set_index('year')

shot_locations_2014 = pd.read_csv(
        'NBA_2014_Shots.csv').rename(columns = {'SEASON_1' : 'year'}).set_index('year')

shot_locations_2004 = pd.read_csv(
        'NBA_2004_Shots.csv').rename(columns = {'SEASON_1' : 'year'}).set_index('year')

top_players_2003_2004 = pd.read_csv('NBA TOP PLAYERS 2003-2004.csv')
top_players_2013_2014 = pd.read_csv('NBA TOP PLAYERS 2013-2014.csv')
top_players_2022_2023 = pd.read_csv('NBA TOP PLAYERS 2022-2023.csv').rename(columns = {'#' : 'rank'})

top_players_2003_2004['year'] = 2004
top_players_2013_2014['year'] = 2014
top_players_2022_2023['year'] = 2023

top_players_2003_2004 = top_players_2003_2004.set_index('year')
top_players_2013_2014 = top_players_2013_2014.set_index('year')
top_players_2022_2023 = top_players_2022_2023.set_index('year')

shot_loc_all_years = pd.concat([shot_locations_2004, shot_locations_2014, shot_locations_2023])

top_players_all_years = pd.concat([top_players_2003_2004, top_players_2013_2014,
                                   top_players_2022_2023])

top_players_shot_loc = shot_loc_all_years.merge(top_players_all_years, how = 'right',
        left_on = ['year', 'PLAYER_NAME'],
        right_on = ['year', 'PLAYER'], )

shot_loc_all_years.to_csv('shot_loc_all_years.csv')
top_players_shot_loc.to_csv('top_players_shot_loc.csv')
top_5_players = pd.read_csv('Top 5 players 2004-2023.csv')
```

## Result: Research Question 1

### Interpretations:

- Position: The most common position is Forward. This is to be expected because the Forward position is basically a hybrid position on the court. Players who play this position basically play every position on the court and move everywhere. This is the most flexible position so this makes sense that the top players are most commonly in this position. It was interesting to see how many players have Forward in their position as this seems to be the most common position. We thought that the center would have more players in that position because they basically scored points under the hoop. But, the forward can do the job as a center and a guard, therefore it is the most hybrid position so this is the most common position of successful players.
- Height: While 6 foot 7 inches was the most common height for the top 5 players in each season, we think that this would change with more data so we can not conclude that this height is what makes players successful. Although, players that were 6 foot 7 and 6 foot 8 made up one third of the top 5 players in the respective seasons. Maybe this will tell us that there is an in between height for not too tall and not short for players to be successful and thrive. That height is a nice hybrid height for players allowing them both height and mobility. This result was a bit surprising in that we thought that the taller players would probably get more points because their job is to stand under the hoop and score, but it seems like between the height of 6 foot 7 and 6 foot 8 inches is a common height for successful players because it is a good in between.
- Weight: Players are most commonly between 200-249 lbs with a mean of about 233 lbs. Players in the NBA are generally taller, and weigh more than the average person therefore this makes sense. All the players in the top 5 seem to be generally close to this weight range. Therefore, we can conclude that players in the top 5 seem to be close to that weight range.



- PPG, APG, RPG: To make a conclusion about these statistics, we concluded that successful players don't just score a high number of points. They also have a high number of rebounds and assists which means they make meaningful contributions to the team. These successful players are more well rounded and affect all aspects of the game, therefore making them successful.

Impacts and Limitations:

- We think that there are a variety of implications of these results. While these results do provide basics to what makes a successful player, there are other players that are successful and don't exactly match these analyses. There are some limitations to our analysis which could include changes to the data from more data from each year or a wider variety of years. This analysis makes broad generalizations about the successful players, and while some of the conclusions may be accurate, they are not very precise which makes that another limitation. Players and team staff may benefit from this data because it shows that the successful players are well rounded in all areas of the game, but being at that level, they probably already know that.

```
In [3]: #position
def player_position(data):
    """
    Creates the plot to show the counts for the number of players in each repective position given the data.
    Parameters:
    - data: DataFrame containing the player data.
    """
    positions = data['POSITION']
    fig = px.bar(data, x = positions, title='Player Positions for Top 5 Players from 2004, 2014, and 2023', color = "POSITION", hover_data="Player Name")
    fig.update_layout(yaxis_title = "# of Players")

    fig.show()

# height
def player_height(data):
    """
    Creates a plot for the distribution of player heights given the data.
    Parameters:
    - data: DataFrame containing the player data.
    """
    fig = px.bar(data, x = "HEIGHT", title="Top 5 Players from 2004, 2014, and 2023 height counts", color = "HEIGHT", hover_data = "Player Name")
    fig.update_layout(yaxis_title = "# of Players")
    fig.show()

# weight
def player_weight(data):
    """
    Creates the plot for the distribution of player weights given the data.
    Parameters:
    - data: DataFrame containing the player data.
    """

    avg_weight = data['WEIGHT'].mean()
    fig = px.bar(data, x="Player Name", y = "WEIGHT", title='Weights for Top 5 Players from 2004, 2014, and 2023', color = "Player Name")
    fig.add_hline(y=avg_weight, line_color="red", annotation_text=f"Mean weight: {avg_weight:.3f}")
    fig.show()

#points, assits, rebounds per game
def player_statistics(data, stat):
    """
    Returns a plot of the distribution and calculates the average for a given statistic. The
    Parameters:
    - data: DataFrame containing the player data.
    - stat: String representing the statistic to analyze ('PPG', 'APG', 'RPG') from the data. Also, we assume the statistic is present in the data.
    """

    avg = data[stat].mean()
    fig = px.bar(data, x='Player Name', y=stat, title=f'{stat} for Top 5 Players from 2004, 2014, and 2023', color = "Player Name")
    fig.add_hline(y=avg, line_color="red", annotation_text=f"Mean {stat}: {avg:.3f}")

    fig.show()
    return avg

player_position(top_5_players)
player_height(top_5_players)
player_weight(top_5_players)
player_statistics(top_5_players, 'PPG')
player_statistics(top_5_players, 'APG')
player_statistics(top_5_players, 'RPG')
```







Out [3]: 7.866666666666665

## Research Question 2

### Interpretations:

- Our results came in the form of Heatmaps that we could qualitatively analyze. We looked at all the made shots and where they were most commonly taken from.

### 2004

No description has been provided for this image

### 2014

No description has been provided for this image

### 2023

No description has been provided for this image

- although our ideal shot map would be a subplot, we are unable to configure the axes and graphs to accommodate multiple jointplots so we have to individually generate it given the specific parameters we need.
- We can see that over time there is an increased density past the 3-point line and just farther from the center of the hoop. While the distribution is still leaning heavily towards the paint area for what shots were made, we can see the difference in how made shots from 3 are occurring more frequently.
- What does this mean? This means that as time goes forward, we can expect Pro basketball to shift more to a game that is decided by the range of players and their playmaking behind the 3. With the rules and regulations of the game's physicality becoming more strict this could be the route that the NBA follows. A lot of the players are perfecting their range whilst still working on their skills and power inside the paint. this is evident from the heatmaps that we see.
- Some implications are definitely the data that we used to generate this graph with. We thought that since the top 50 players, ranked by the NBA themselves, would prove to be the players who make the most difference in a game, they would be a good candidate for graphing a heatmap with. This is also largely because to generate the heatmap for all of the NBA, our code would have to create a graph for 200,000+ lines rows. This was possible with the scatterplot version but it was very very cluttered and would not load for the heatmap version so we had to compromise our data to only the top 50 players in that year. In conclusion, our graphs are generated with the idea that the top 50 players would represent all of the NBA's shot trends as with our analysis being in the same boat.

```
In [4]: '''
reference source:
https://github.com/bradleyfay/py-Goldsberry/blob/master/docs/
Visualizing%20NBA%20Shots%20with%20py-Goldsberry.ipynb
'''
def draw_court(ax = None, color = "black", lw = 2):
    '''
    This method draws a basketball court that is up to the NBA regulation dimensions. dimensions
    from https://www.nba.com/stats . takes in an axes to draw it on, the color of the lines, and
    the linewidth. Returns an axes with the court, drawn to scale.
    '''
    if ax is None:
        ax = plt.gca()

    outline = Rectangle((-25, 0), width=50, height=47, fill=False, linewidth=lw, color=color)

    hoop = Circle((0, 65/12), radius=0.75, fill=False, linewidth=lw, color=color)

    backboard = Rectangle((-3, 4), width=6, height=0, linewidth=lw, color=color)

    in_center_box = Rectangle((-6, 0), width=12, height=19, fill=False, linewidth=lw, color=color)

    out_center_box = Rectangle((-8, 0), width=16, height=19, fill=False, linewidth=lw,
                               color=color)

    ft_top = Arc((0, 19), width=12, height=12, theta1=0, theta2=180, fill=False, linewidth=lw,
                 color=color)

    ft_bottom = Arc((0, 19), width=12, height=12, theta1=180, theta2=0, fill=False, linewidth=lw,
                    color=color, linestyle="dashed")

    three_point_arc = Arc((0, 65/12), 47.5, 47.5, theta1=22, theta2=158, linewidth=lw,
                          color=color)

    three_point_side1 = Rectangle((-22, 0), width=0, height=14.3, linewidth=lw, color=color)

    three_point_side2 = Rectangle((22, 0), width=0, height=14.3, linewidth=lw, color=color)

    restricted_zone = Arc((0, 65/12), width=8, height=8, theta1=0, theta2=180, linewidth=lw,
                          color=color)

    half_court_in = Arc((0, 47), width=4, height=4, theta1=180, theta2=0, linewidth=lw,
                       color=color)

    half_court_out = Arc((0, 47), width=12, height=12, theta1=180, theta2=0, linewidth=lw,
                        color=color)

    court_elements = [outline, hoop, backboard, in_center_box, out_center_box, ft_top, ft_bottom,
                      three_point_arc, three_point_side1, three_point_side2, restricted_zone,
                      half_court_in, half_court_out]

    for element in court_elements:
        ax.add_patch(element)

    return ax
```

```
In [5]: def shot_chart(player_name='All Players', year=None, date=None, quarter=None, data=None, ax=None, all_shots:
    '''
    This method draws the shot chart from a dataset that contains shot locations. Personalizable
    by the player, year, date of game, quarter, Only displaying made shots, and an option between
    scatterplot and heatmap. If no player is named, it shows all players. if no year, date, or
    quarter is specified, shows all of them. By default, it shows all shots and is a scatterplot.
    '''
```



```
if ax is None:
    ax = plt.gca()

shot_loc = data

if player_name != 'All Players':
    shot_loc = data[data['PLAYER_NAME'] == player_name]

if year != None:
    shot_loc = shot_loc.loc[year]
else:
    shot_loc = shot_loc.loc[:]

if date != None:
    shot_loc = shot_loc[shot_loc['GAME_DATE'] == date]
if quarter != None:
    shot_loc = shot_loc[shot_loc['QUARTER'] == quarter]

if all_shots == False:
    shot_loc = shot_loc[shot_loc['SHOT_MADE'] == True]

if scat_heat == 'heat':
    #source reference: http://savvastjortjoglou.com/nba-shot-sharts.html
    cmap=plt.cm.YlOrRd_r

    joint_shot_chart = sns.jointplot(shot_loc, x = 'LOC_X', y = 'LOC_Y',
                                     kind='kde', space=0, color=cmap(0.1),
                                     cmap=cmap, n_levels=50)

    joint_shot_chart.fig.set_size_inches(7,6)
    ax = joint_shot_chart.ax_joint
    ax.set(xlabel=None, ylabel=None, title=f"{player_name} {year} shot chart")

else:
    markers = {True : 'o', False : 'X'}
    sns.scatterplot(shot_loc,
                    ax=ax, x = 'LOC_X', y = 'LOC_Y', style = 'SHOT_MADE', hue = 'SHOT_MADE', markers=markers)
    ax.set(xlabel=None, ylabel=None, title=f"{player_name} {year} shot chart")
    ax.legend(title="Shot Made")

ax.set_aspect(1)
ax.set_xlim(-28, 28)
ax.set_ylim(-3, 50)
draw_court(ax)

return ax
```

Research Question 3

Interpretations:

- 3P%: For the 2003-2004 season the mean 3P% was 31.3, for the 2013 - 2014 season it was 32.616, and for 2022-2023 it was 35.482. This indicates that the average three-point percentage is increasing over time. A conclusion from this could be that players are getting better at shooting three point shots, therefore the average three point percentage is increasing. This was something that we initially predicted but it was definitely interesting to see verified in our analysis. We thought initially that there has been a shift in players shooting more three point shots over the years, and are getting more accurate in their shots as well. This was just a prediction based on watching the game over the years, but since doing the analysis, it was very interesting to see it present in the data. Therefore, we can conclude that players are getting more accurate at shooting 3 point shots over the years.
- FG%(2-points and 3-points): For two point shooting, we see that the average percentage for 2003-2004 is 45.148, for 2013-2014 it is 45.922, and for 2022-2023 it is 47.71. Once again, this indicates that players are on average making more shots and being more accurate shooters over time. We did not expect this to be the case, because this is basically total shots made. However, after analyzing the data more this makes sense as it basically shows that players over time are improving overall and getting better at most individual aspects of the game.
- PTS: For the average points made in a season, the average was 17.95 in 2004, 18.35 in 2014, and 23.318 in 2023. This also indicates that individual players are scoring more points over time. From 2004 to 2023, there is a large jump of about 6 points per game more in 2023 than in 2004. This was interesting to see because it shows how the game has evolved in that players are scoring more points per game than before.
- REB: For the average number of rebounds over time, the average has decreased over time. In 2004 the average number of rebounds was 6.444, in 2014 it was 5.802, and in 2023 it was 5.8. This was a shocking result at first, but as we thought about it we think our other data can help explain this result. A possible explanation is that players are making more shots than missing, therefore there are less rebound opportunities in a game. This is shown in our other data because the shot percentages have increased and the rebounds have decreased which makes sense because there are less rebounds to be made if more players are making more shots.
- APG: The assists per game also increased over time. In 2004 the mean assists per game was 3.724, for 2014 it was 4.306, and for 2023 it was 5.042. The assists per game are increasing which was not expected. We did not think that

assists would show any correlation over time, but an explanation for this is also supported by our shooting percentage analysis. Because players are making more shots and more points, this leaves the players with more opportunities to get assists than before. Therefore, there are more assists over time.

Impacts and Limitations:

- A limitation of this data is that this only reflects the data for the top 509 players for each year, so we can not generalize what the stats or trends are for every player in that season. Another limitation is that we only did three years in about 10 year increments so there could have been outlier years that could have shown differences that our results. Although this shows a general trend, this might not show a positive correlation from back to back years as those may differ because we once again used years in around 10 year increments. We think that players and coaches can use this data and focus in on areas of the game that have more opportunities to be improved. Also, the recognition that players just keep improving over time is helpful to know for training players.

```
In [6]: def stat_changes(data_one, data_two, data_three, stat):
        """
        Plots the figure which is three box plots, one for the 2003-2004 season,
        one for the 2013-2014 data, and one for the 2022-2023 data, which shows the distribution
        for the the string statistic that is given. We are also assuming that the stat is present in the data.

        """
        fig = go.Figure()

        fig.add_trace(go.Box(y=data_one[stat], name='2003-2004', marker_color='blue', boxmean=True))
        fig.add_trace(go.Box(y=data_two[stat], name='2013-2014', marker_color='orange', boxmean=True))
        fig.add_trace(go.Box(y=data_three[stat], name='2022-2023', marker_color='green', boxmean=True))

        if stat == '3P%' or stat == 'FG%':
            yaxis_title = f"{stat} Percentage"
        elif stat == 'PTS':
            yaxis_title = "Points Per Game"
        else:
            yaxis_title = stat

        fig.update_layout(
            title=f"{stat} changes between 2004, 2014, and 2023 seasons",
            yaxis_title=yaxis_title,
            xaxis_title="Season",
            height=800,
            width=1000
        )
        fig.show()

stat_changes(top_players_2003_2004, top_players_2013_2014, top_players_2022_2023, '3P%')
stat_changes(top_players_2003_2004, top_players_2013_2014, top_players_2022_2023, 'FG%')
stat_changes(top_players_2003_2004, top_players_2013_2014, top_players_2022_2023, 'PTS')
stat_changes(top_players_2003_2004, top_players_2013_2014, top_players_2022_2023, 'REB')
stat_changes(top_players_2003_2004, top_players_2013_2014, top_players_2022_2023, 'AST')
```











## Research Question 4

### Interpretations:

- Our results are in the form of a 3x4 subplot for all 4 quarters over the 3 selected years. We matched it up with 12 separate heatmaps for each player's scatter plot. This allows us to see both scatter and heatmap so we can make inferences between quarters/years. The heat maps are in the same order as the scatter.

### Result


- Since again, this is a map we are looking at and not a number, we have to analyze this qualitatively. We did this by comparing each player to the next and to the same quarter throughout the years. Some things we noticed:
  - As the years progress, at least within our data range, there seems to be a larger spread of players who perform the best on average for every quarter. Again this data is only within the top 50 players for that year so it is missing a ton of other potential players but these were the best on average out of the best 50 players.
  - We noticed that there are multiple players whose specialties are certain ranges. These graphs show how they excel in these ranges and areas during their best quarters. For example, in 2004 Tracy McGrady seemed to dominate in shots made during quarters 2, 3, and 4. His heatmap and scatterplot show during these quarters that he was especially dangerous in the mid and 3-point range. As well as Stephen Curry in 2014, one of the best players in terms of accuracy and consistency from long-range. His Heatmap shows this and how versatile his game is between long-range and in the paint with layups. There are also players like Luka Doncic and Ja Morant in 2023 who excel in close-quarters shots and mid-range. Of course, this is all dependent on the player that is calculated a best performing per that quarter but these were the players that we had resulted with.










### Limitations

- We really wanted to try to show our results with only the heat maps but attempt after attempt at trying to get the joint plot to work correctly with subplots, we settled with our results now which are acceptable for us.
- Our calculations for who performs the best was very limiting for who got chosen as the best for that quarter and year. We went with who scored the most points for that quarter out of all of that specific quarter that was played by that player. Basketball is a team game and this code only looks at the points. It ignores the effort made by the other teammates who perhaps got their teammates a score whilst they only got an assist, rebound, block, etc. We Understood this but there

wasn't enough time and knowledge that we had to figure out a function that could take all of this into account so we were forced to settle with solely points.

- Another thing to consider is that these maps and calculations consider all games in the season. This includes Pre-season, where many starter players don't play as much or put in as much effort to avoid injury before their regular season games. Also, some players will play more games depending on their team's standing in the playoffs and regular season, which could skew the data.

No description has been provided for this image

```
In [12]: def performs_best_quarter(year, quarter, data):
    """
    Returns the name of the player that has the highest score per 'quarter' quarter played. takes
    in a specific year and quarter and a dataset to look over.
    """
    specific_quarter = data[(data['QUARTER'] == quarter) & (data['SHOT_MADE'] == True)].loc[year]
    player_game_count = specific_quarter.set_index('PLAYER_NAME').GAME_ID
    player_scores = {}

    for player, type in zip(specific_quarter['PLAYER_NAME'], specific_quarter['SHOT_TYPE']):

        if player not in player_scores:
            player_scores[player] = 0

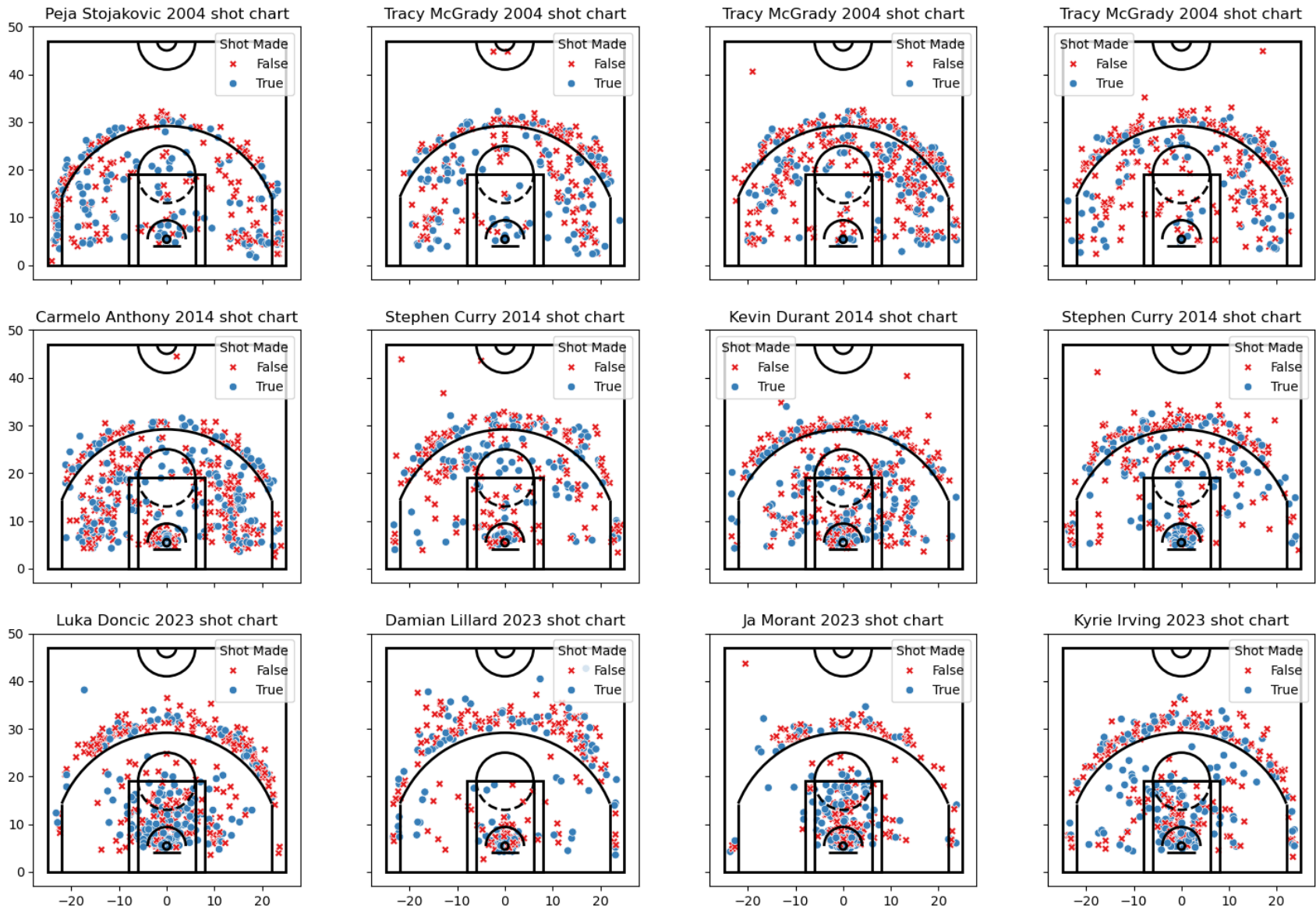
        if type == '3PT Field Goal':
            player_scores[player] += 3
        elif type == '2PT Field Goal':
            player_scores[player] += 2

    player_avg_score = {}
    for player in player_scores.keys():
        games = player_game_count.loc[player].tolist()
        if not isinstance(games, list):
            games = 1
        else:
            games = len(set(games))
        player_avg_score[player] = player_scores[player] / games

    return max(player_avg_score, key=player_avg_score.get)
```

```
In [8]: """
This is meant to show the shot chart for all 12 quarters over the three years in a 3x4 subplot.
"""

fig, axs = plt.subplots(nrows = 3, figsize = (18, 12), ncols = 4, sharex = True, sharey = True)
years = [2004, 2014, 2023]
quarters = [1, 2, 3, 4]
for year in years:
    for quarter in quarters:
        shot_chart(performs_best_quarter(year, quarter, top_players_shot_loc), year = year,
                    quarter = quarter, data = top_players_shot_loc, ax = axs[years.index(year),
                    quarters.index(quarter)], all_shots = True)
    fig.savefig('best_perform_shot_loc.png')
```



### Challenge Goals

- From our proposal we initially thought to make a machine learning model that would predict how a certain player would perform against a given team, or other conditions. We realized that neither of us really knew and were confident enough to implement that so we reevaluated our challenge goals. Our new challenge goals were:
  - Messy Data: We webscraped data from NBA Stats for various and specif years and players. Three of our data sets are the top 50 players from 2004, 2014, and 2023 seasons which were we scraped from NBA Stats. Another one of our data sets is top 5 players from those years that ocntian more data such as phycial characteristics, country, and draft pick. We did this extra scraping to the top 5 playuers only because it would have taken too long for us to scrape this for 150 players total beacause we had to indiuidally.
  - New Library: We used plotly for some of our visualizations as one of our challenge goals. This was a new library for us because we had not used it in class before. This allowed us to make our plots interactive as it shows more information when you hover over it, and can zoom in and scale the plot differently. This was a challenge because we had not really had to learn something new like this by reading the documentation but although it is not perfect, we got better at it and learning it. We used both plotly.express and plotly.graph\_objects for this. We had to use both because for one of the tasks, we wanted to show multiple histograms on one plot. So, graph\_objects is better for this so we used that for research question 3. It was interesting to learn all the capabilities of plotly and what we only used is a small fraction of the differetrn visualizations plotly can make.
  - Multiple datasets: For one of our analysis we used a merge operation to merge our datasets together to have a data set that allowed us to access the top player's stats and their shot locations every year.

### Plan Evaluation

- We changed a lot of our initial proposal after going over our questions and receiving feedback. We decided to web scrape data, not use the machine learning algorithm, and changed most of our research questions.
- For most of this, we underestimated the time it took for almost everything. Web scraping took about 3 hours to gather the data, the coding took about 5 hours each as well because a lot of it was looking up in the documentation/stack exchange for answers. Then writing our analysis took about 4 hours, to make conclusions about our data.

### Testing

- For testing, we were very unsure on how to test our code from the visualizations. The only thign we thought of was testing the mean calcualtion to make sure that it was working. Though, this test seems unecessary becuae it seems that we are basically jsut testing the .mean() function. Here we just added a return to the player\_statistics method which is the calcualted mean. Then here we are just checking to see if that calcualtion matches the mean that is to be expected.

```
In [9]: expected_mean_ppg = top_5_players['PPG'].mean()
expected_mean_apg = top_5_players['APG'].mean()
expected_mean_rpg = top_5_players['RPG'].mean()

test_mean_ppg = player_statistics(top_5_players, 'PPG')
test_mean_apg = player_statistics(top_5_players, 'APG')
```

```
test_mean_rpg = player_statistics(top_5_players, 'RPG')

assert test_mean_ppg == expected_mean_ppg, "PPG mean calculation is incorrect"
assert test_mean_apg == expected_mean_apg, "APG mean calculation is incorrect"
assert test_mean_rpg == expected_mean_rpg, "RPG mean calculation is incorrect"
```



### Collaboration

- For this project, we used the plotly documentation: <https://plotly.com/python/> . we used both graph\_objects and express.
- We used TA feedback
- We searched online for a lot of the graphing information and documentation. GitHub, Reddit, Stack Exchange, and other useful websites were used for information on how to do.
  - <https://seaborn.pydata.org/api.html> and <https://matplotlib.org/stable/api/index.html> were used greatly for graph production.
  - <https://github.com/bradleyfay/py-Goldsberry/blob/master/docs/Visualizing%20NBA%20Shots%20with%20py-Goldsberry.ipynb> for information on drawing the basketball court and

In [ ]: