Sejal Dua
Megan Monroe
COMP152: Sports Analytics
23 March 2021

SPORT VU LAB

NBA tracking data can be very messy, but it is an asset that can quickly become valuable after a bit of preprocessing. The applications of Sport VU data are truly endless; it can be used to diagnose why a given team lost a specific game, or it can be used to see how much space defenders give Damian Lillard, on average, depending on how far away he is from the hoop (spoiler alert: don't give that man any space!)... or really anything in between. The biggest challenge with this mammoth amount of data is wrangling it into a usable format. Tracking data for each game includes the following:

- *position table*: contains the (x, y) position of very player on the court and the (x, y, z) coordinates of the ball, at 25 frames per second
- *event table*: contains play-by-play data, i.e., a chronological list of events that occur in the game, and when each event occurred
- *video*: live game footage

The purpose of this lab is to use position and event data to come up with heuristics for identifying shots in a 2016 basketball game between the Brooklyn Nets and the Boston Celtics. Once we have identified shots, we will plot an interesting and informative shot fact with respect to time to see if we can derive any insights regarding shot selection over the course of the game.

Cleaning and Wrangling the Data

Loading and preprocessing the data was, by far, the most difficult part of this lab. Thanks to powerful Pandas operations, it was not too difficult to merge the two sources of data and go from there. First, the event data was read into a Pandas DataFrame, and the first row was discarded. Next, the SportVU tracking data was also read into a Pandas DataFrame, and the event ids were retrieved using an apply function. The event data was filtered by the "EVENTMSGTYPE" column to only include events corresponding to shot attempts (makes and misses). Then, the shot event data was merged with the tracking data on the event id column to make sure that the correct moments were being aligned with the play-by-play events. This merged dataset is previewed below, where the tracking data exists in dictionary format in the rightmost column.

	EVENTNUM	EVENTMSGTYPE	PERIOD	PCTIMESTRING	HOMEDESCRIPTION	VISITORDESCRIPTION	PLAYER1_ID	events
0	2	1	1	11:42	NaN	Young 1' Reverse Layup (2 PTS) (Johnson 1 AST)	201152	{'eventId': '2', 'visitor': {'name': 'Brooklyn...
1	4	2	1	11:14	NaN	MISS Young 21' Jump Shot	201152	{'eventId': '4', 'visitor': {'name': 'Brooklyn...
2	9	1	1	10:44	NaN	Lopez 8' Fadeaway Jumper (2 PTS) (Jack 1 AST)	201572	{'eventId': '9', 'visitor': {'name': 'Brooklyn...
3	10	2	1	10:29	MISS Thomas 25' 3PT Jump Shot	NaN	202738	{'eventId': '10', 'visitor': {'name': 'Brookly...
4	12	2	1	10:21	Sullinger BLOCK (1 BLK)	MISS Bogdanovic 4' Jump Shot	202711	{'eventId': '12', 'visitor': {'name': 'Brookly...
...	...	...	...	...	...	...	...	...
176	582	1	4	0:26	NaN	Lopez Dunk (30 PTS) (Young 1 AST)	201572	{'eventId': '582', 'visitor': {'name': 'Brookl...
177	585	2	4	0:23	MISS Crowder 24' 3PT Jump Shot	NaN	203109	{'eventId': '585', 'visitor': {'name': 'Brookl...
178	588	2	4	0:14	MISS Thomas 27' 3PT Jump Shot	NaN	202738	{'eventId': '588', 'visitor': {'name': 'Brookl...
179	590	1	4	0:11	Turner 2' Putback Layup (12 PTS)	NaN	202323	{'eventId': '590', 'visitor': {'name': 'Brookl...
180	595	2	4	0:02	MISS Jerebko 3PT Jump Shot	NaN	201973	{'eventId': '595', 'visitor': {'name': 'Brookl...

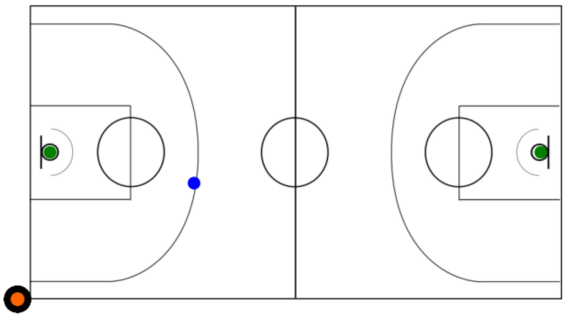
181 rows × 8 columns

As an auxiliary data source, I also put together a little DataFrame and dictionary in case player information became a relevant heuristic later.

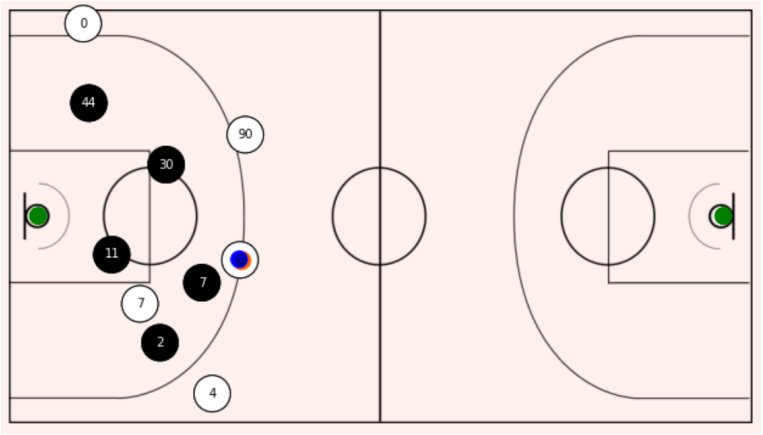
	lastname	firstname	playerid	jersey	position
0	Bradley	Avery	202340	0	G
1	Thomas	Isaiah	202738	4	G
2	Sullinger	Jared	203096	7	C-F
3	Jerebko	Jonas	201973	8	F
4	Turner	Evan	202323	11	G-F
5	Young	James	203923	13	G-F
6	Smart	Marcus	203935	36	G
7	Olynyk	Kelly	203482	41	C-F
8	Lee	David	101135	42	F-C
9	Zeller	Tyler	203092	44	C-F
10	Johnson	Amir	101161	90	F
11	Crowder	Jae	203109	99	F
12	Larkin	Shane	203499	0	G
13	Jack	Jarrett	101127	2	G
14	Johnson	Joe	2207	7	F-G
15	Bargnani	Andrea	200745	9	C-F
16	Karasev	Sergey	203508	10	G-F
17	Lopez	Brook	201572	11	C
18	Sloan	Donald	202388	15	G
19	Ellington	Wayne	201961	21	G
20	Brown	Markel	203900	22	G
21	Young	Thaddeus	201152	30	F
22	Reed	Willie	203186	33	F-C
23	Robinson	Thomas	203080	41	F
24	Bogdanovic	Bojan	202711	44	G-F

Finding Basket Coordinates

For each event, shot clock resets were used to figure out when the ball likely went through the net. Since we know that the rim of a regulation NBA hoop is 10 feet off of the ground, the z coordinate of the ball was used to iterate through the 30 frames leading up to the shot clock reset and figure out the moment at which the z coordinate exceeded 10. At that exact frame of the tracking data, the x and y coordinates of the ball should theoretically align with the x and y coordinates of the basket. To be more robust with this, each shot event id was fed into a function called find_basket_coords, and the returned x and y estimates for the basket coordinates were averaged across all calls to this function. One basket was placed at (6.13, 25.25) while the other was placed at (88.08, 25.5). To sanity check these findings, green dots were plotted on the figure below, which is an overlay of a basketball court PNG image.



Additionally, I borrowed / repurposed some of Dan Vatterott's Matplotlib animation code from his blog post, "Creating Videos of NBA Action with Sportsvu Data," in order to have a visual aid rather than blindly debugging and refine my heuristics later. These animations also helped to make sure my data wrangling actually worked (e.g. the animation for event 62 should actually depict Jae Crowder drilling a 3PT jump shot).



Shot Clock Usage

For my heuristic (a.k.a. “shot fact”), I decided to explore shot clock usage. I thought it would be an interesting shot fact to explore because it is obvious that game tempo is a huge factor that influences the nature of an NBA game. However, play-by-play game data makes it quite difficult to infer how often fast breaks are being executed versus settled offense scoring possessions. For clarity, shot clock usage refers to how many seconds of the 24-second shot clock are used before a team shoots the ball. For example, if Steph Curry fires up a shot with 1 second left on the shot clock, the shot clock usage for that possession is 23 (seconds).

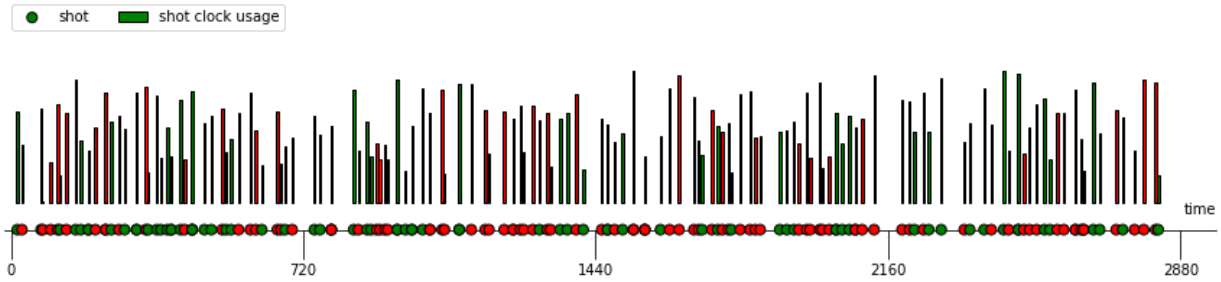
My hypothesis was that the first and third quarters would consist of higher shot clock usage whereas the latter part of the second and fourth quarters would consist of lower shot clock usage. My rationale for this hypothesis is that the game typically intensifies when a team is down and needs to chip away at the score differential with quick buckets.

The actionability of this heuristic lies in the notion that knowing when to settle the offense and knowing when to capitalize on fast breaks is a simple enough strategy to take control of any game. If the data can show that when a team does find that “one more” pass, they are successful, that is a valuable enough insight to bring it to a team’s attention so that they can use it to their advantage.

Pseudocode Algorithm

- get shot clock data for all moments in the specified shot event
- find the moment index at which the shot happened (just prior to when the shot clock resets to 24 seconds)
- at this moment index, get the time left in the quarter
- get quarter number from SportVU moment data (0th index)
- compute the shot time (in milliseconds) with respect to the whole game
    - past elapsed time = (quarter number - 1) * 720
    - current elapsed time = (720 - time left)
    - shot time = past elapsed time + current elapsed time
- compute shot fact (i.e. shot clock usage)
    - shot fact = 24 - time left in quarter at the moment the shot was taken

Shot Timeline



The graph above depicts shot clock usage with respect to game time (in milliseconds). The bars in the bar chart are color code by whether the shot was a make (green) or miss (red).

Assumptions & Limitations

One of the huge assumptions / limitations that this analysis relies on is the shot clock being reset in a pretty consistent and timely manner after each shot has been taken. Given that the shot clock is controlled by a human, a variable lag would make these results more difficult to interpret. That said, if there is a consistent lag due to human reflex times, it should not impact the visualization in any significant way. I also struggled quite a bit in figuring out how to handle events with multiple shots taken. The play-by-play event data and tracking data didn't line up perfectly-- I noticed that some events (e.g. 61) had two shots taken in the same chunk of tracking data. This made it difficult to figure out which shot to keep since keeping both would have screwed up my color-coding scheme which relied upon whether the event type recorded in the play by play data denoted a make or miss. I sorted this issue out by using the player ID from the play by play data and matching that up with the player who last held the ball at the moment the shot took place. There are still a handful of cases which are just too messy to sort out, but that's how the story usually goes with this type of data. In general, my hypothesis is not quite validated. Interestingly, though, towards the start of the second quarter, there was a little streak of fast breaks. The fourth quarter also has a decent amount of short possessions.

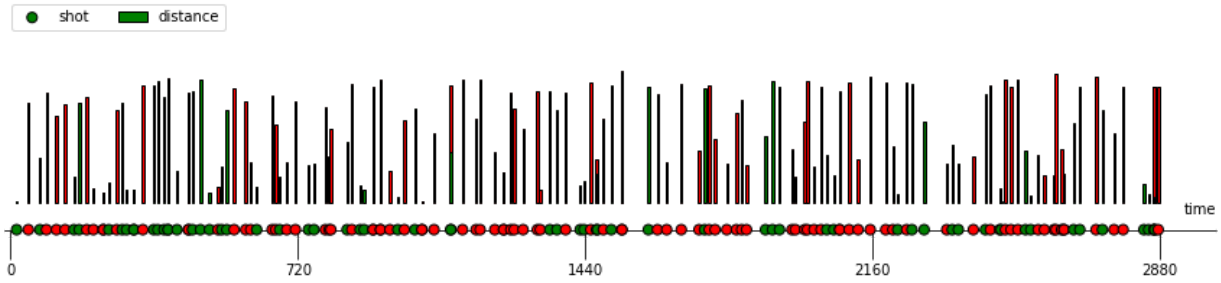
Shot Distance

Just for fun, I also decided to explore shot distance from the hoop because I thought it would be disrespectful to the position data if I didn't at least attempt some geometry with it.

Pseudocode Algorithm

- get ball xyz and player xy coordinates from the specified events
- instantiate array of possession moments
- loop through all moments
    - loop through all players on the floor
        - if the euclidean distance between current player and the ball is less than or equal to 2.5 and the ball's z coordinate is less than 9
            - add the player jersey number and player xy coordinates to array of possession moments
        - if the euclidean distance between the ball and basket 1 is less than or equal to 3 OR the euclidean distance between the ball and basket 2 is less than or equal to 3 AND the ball's z coordinate is above 9.5
            - reverse iterate through possession moments
                - if the jersey number matches the player who shot the ball from the play-by-play data
                    - get the quarter number
                    - get the time left in the quarter
                    - compute shot time (same as before)
                    - compute shot fact (euclidean distance between the position the player was at when the ball left their hands and the basket)

Shot Timeline



From the above graph, we can notice that there were many cold streaks during the game, but one noteworthy shooting period was in the middle of the first quarter when there was a back-and-forth contest of 3PT shots between Johnson and Crowder. As one might imagine, the majority of the shorter bars represent made baskets whereas the majority of the longer bars represent missed 3PT attempts.

Limitations

Putback dunks were incredibly hard to detect for this analysis. I hypothesize that because my algorithm relies on the ball's z coordinates not being anywhere near 10, putback dunks cannot be captured since the ball never comes down.

General Reflections

This was a very fun lab. I plan to make more GIF animations of each play and also look at distance between shooter and closest defender as another heuristic! I really wonder why the NBA revoked this data from public access... that is super not chill of them.