NBA Team Depth Analysis

Andrew Edelman

2022-10-12

Contents

Objective

The objective of this analysis is to use team depth as a measure of determining the best and worst teams in the NBA. Depth is roughly defined by how many "good" players a team has. Teams in the NBA start 5 players, and usually play ~8-10 players during a regular season game. A team with good depth would be described as a team that has "good" players not only in their starting 5, but also going in to the 6-8 roster spots.

While it shouldn't be considered a sole means to predict championship winners, depth is generallly regarded as a mark of a good/bad team, and can be used to identify serious contenders, and the bottom of the barrel.

The goal here is to determine each team's depth by categorizing players using cluster analysis. Players will be categorized by a variety of statistics from the previous seasons (explained in more detail later on), and assigned a value scored based on the cluster they land in; essentially a higher score representing a better player. Value scores at the player level will then be used to determine overall team "depth value", to identify top and bottom teams. We'll then compare this result to the official nba.com power ranking to see how depth rankings stack up against it.

Load in packages used

```
library(dplyr)
library(ggplot2)
library(tibble)
library(tidyr)
library(readr)
library(knitr)
library(tinytex)
library(formatR)
```

Load in Data

Data comes from basketball-reference.com per game player stats for the 2021-22 NBA season

```
# read in data, saved locally via the basketball-reference
# link
player_data_raw <- read_csv("player_20212022_per_game_stats.csv")

# check column data types
str(player_data_raw)
```

About the data

This data contains each player as an "observation". Looking at the tibble structure, column types have imported correctly. The only non-numeric columns in the dataframe include: Player (string, player name), Pos (string, player position), and Tm (string, player team). All other columns represent some numeric statistic, that was that player's per-game average over the course of the 2021-22 season. For example, the "PTS" column represents average per-game points for that player.

Clean Data

Identify Duplicate Rows

Looking at the bball reference data ahead of time, it's important to note that an observation is created per-player, per-team, and a "Total". Meaning that if a player played for three teams in the 2021-22 season, they have four entries in the dataframe. One for each team they played on, plus a "total" season average row. However, the actual stats are running averages through season, going from most recent team (team they ended the season with), to least recent (team they started the season with). So ultimately we just need to keep the "total" observation for each player to get their end-of-season stats, since that takes in to account their whole season, regardless of teams played for.

We can see below that this holds up, where n represents the number of observations for that player in the table, AKA the number of teams they played for in 2021-22, plus 1 (the total row). For players who only played for one team the entire season, there is no "total" row, so no duplicate handling is needed.

```
# first we'll validate this by checking for duplicate
# player entries
player_data_raw %>%
    count(Player) %>%
    filter(n > 1)
```

```
## # A tibble: 97 x 2
##    Player             n
##    <chr>          <int>
##  1 Aaron Holiday      3
##  2 Alize Johnson      4
##  3 Andre Drummond     3
##  4 Armoni Brooks      3
##  5 Brad Wanamaker     3
##  6 Braxton Key        3
##  7 Bruno Fernando     3
##  8 Bryn Forbes        3
##  9 Buddy Hield        3
## 10 Cam Reddish        3
## # ... with 87 more rows
```

Remove Duplicate Rows

We'll remove all rows for players who played for more than one team, keeping only the "Total" row (the row where team = "TOT").

```
# check for dupes by player name and rank, where rank is a
# unique identifier per-player, just in case there are
# players with the same name our data is already set up so
# that we only want to keep the first duplicate row, so
# distinct() will work easily here store this in a new
# cleaned dataframe

player_data_nodupes <- player_data_raw %>%
    distinct(Rk, Player, .keep_all = TRUE)

# run a distinct count by rank (unique player id) to make
# sure we didn't drop any data accidentally
n_distinct(player_data_raw$Rk)
```

```
## [1] 605
```

```
n_distinct(player_data_nodupes)
```

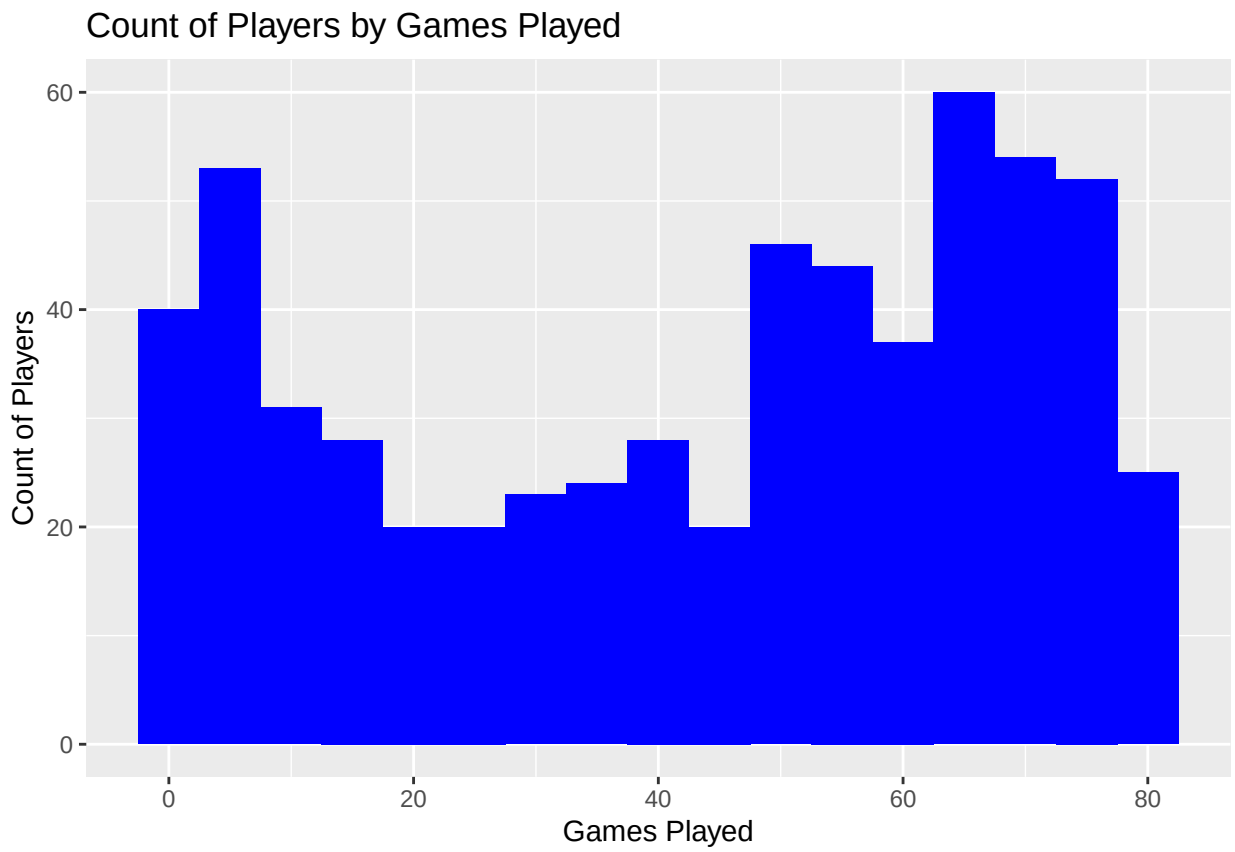
```
## [1] 605
```

Remove players with few games

We'll also want to remove players who only played a few games throughout the season. In general, these are fringe-NBA players or guys who are way down in the bench and only getting minutes due to injuries to

regular rotation pieces. The "lowest level" cluster we're interested in is the "garbage time" cluster, but still only taking in to account players are regularly getting those minutes throughout the season, not just for a few games.

```
player_data_nodupes %>%
    ggplot(aes(x = G)) + geom_histogram(binwidth = 5, fill = "blue") +
    xlab("Games Played") + ylab("Count of Players") + ggtitle("Count of Players by Games Played")
```



Looking at the histogram, there's a secondary peak of players who played 0-15 games in the 2021-22 season, we'll aim to drop players with 15 or fewer games. Let's first look at the players who will be dropped.

```
# filter the removed-dupe dataframe for players with fewer
# than 15 games
player_data_nodupes %>%
    filter(G <= 15) %>%
    arrange(desc(GS))
```

```
## # A tibble: 144 x 31
##       Rk Player      Pos     Age Tm        G    GS    MP    FG   FGA `FG%`  `3P`
##    <dbl> <chr>       <chr> <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1   223 Joe Harris  SF       30 BRK      14    14  30.2   4     8.9 0.452   2.9
##  2   336 Brook Lopez C        33 MIL      13    11  22.9   4.7  10.1 0.466   1.5
##  3   498 Collin Sex~ SG       23 CLE      11    11  28.7   6.2  13.7 0.45    1
##  4   452 Michael Po~ SF       23 DEN       9     9  29.4   4.1  11.4 0.359   1.1
##  5   243 Jaylen Hoa~ SF       22 OKC       7     5  34.3   6.3  12.9 0.489   1.3
##  6   302 Georgios K~ SF       23 TOT      13     4  16.5   2.4   5.2 0.463   0.8
```

```
##  7   505 Zavier Sim~ PG       24 OKC       4     4  43.5   4.8  13   0.365   0.3
##  8   135 Mamadi Dia~ PF       25 OKC      13     3  14.5   1.9   3.6 0.532   0
##  9   153 Kris Dunn   PG       27 POR      14     3  24     3.1   7.3 0.431   0.1
## 10   177 Tim Frazier PG       31 TOT      12     3  17.3   1.2   3.8 0.311   0.5
## # ... with 134 more rows, and 19 more variables: '3PA' <dbl>, '3P%' <dbl>,
## #   '2P' <dbl>, '2PA' <dbl>, '2P%' <dbl>, 'eFG%' <dbl>, FT <dbl>, FTA <dbl>,
## #   'FT%' <dbl>, ORB <dbl>, DRB <dbl>, TRB <dbl>, AST <dbl>, STL <dbl>,
## #   BLK <dbl>, TOV <dbl>, PF <dbl>, PTS <dbl>, 'Player-additional' <chr>
```

This results in 144 players of our total of 605 players who would be dropped. And here's where it gets a little tricky - we'll want to make sure we do not drop any notable players who may have played fewer than 15 games due to injuries, and are expected to return for the upcoming season, and who also played enough games to get a valid stat line.

GS (Games Started) seems to be a good variable to use here. If a player has a notable number of games started, it's likely they are not a fringe NBA player, and got injured early in the season, and will be returning in the upcoming season. Looking through the data, this would include players like Collin Sexton (11/11 games started), Michael Porter Jr. (9/9 games started), Brook Lopez (11/13 games started), and Joe Harris (14/14 games started).

Note: this dataset does not include players who played 0 games in the 2021-22 season. There are many highly impactful players who will be returning in the 2022-23 season not in this dataset, such as Kawhi Leonard, Jamal Murray, Zion Williamson, etc. These players will be handled separately later on.

For now, we'll drop all players who played 15 or fewer games, unless they started in at least 5 games.

```
# drop players with 15 or fewer games played, unless they
# started in at least 5 games

player_stats_games_played <- player_data_nodupes %>%
    filter(GS >= 5 | G > 15)

# run count on remaining data to gut check remaining
# observations
count(player_stats_games_played)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   466
```

We started with 605 players, and from above, we know that 139 were to be dropped (the total of 144 who played 15 or fewer games, minus the 5 who started in at least 5 games). 605 - 139 = 466, so this checks out.

Keep only variables of interest

Next, we'll filter down the cleaned dataframe to keep only the variables of interest for clustering. F or this analysis, we'll use a relatively standard statline includeing Points Per Game, Rebounds Per Game, Assists Per Game, Steals Per Game, Blocks Per Game, and Effective Field Goal Percentage.

```
# select appropriate columns
player_stats_final <- player_stats_games_played %>%
    select(Rk, Player, PTS, TRB, AST, STL, BLK, eFG = "eFG%")
head(player_stats_final)
```

```
## # A tibble: 6 x 8
##      Rk Player                    PTS   TRB   AST   STL   BLK   eFG
##   <dbl> <chr>                   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1 Precious Achiuwa          9.1   6.5   1.1   0.5   0.6 0.486
## 2     2 Steven Adams              6.9  10     3.4   0.9   0.8 0.547
## 3     3 Bam Adebayo              19.1  10.1   3.4   1.4   0.8 0.557
## 4     4 Santi Aldama              4.1   2.7   0.7   0.2   0.3 0.424
## 5     5 LaMarcus Aldridge        12.9   5.5   0.9   0.3   1   0.566
## 6     6 Nickeil Alexander-Walker 10.6   2.9   2.4   0.7   0.4 0.449
```

Add data for missing/new players

Injured players who didn't play in 2021-22

As mentioned above, there are a few players returning for the 2022-23 season who did not play in 2021-22, and are missing from our dataset so far. We'll want to include these players in the analysis as they'll have a significant impact on their team's depth rating. While this isn't an all-encompassing list, some of the highest impact players returning in 2022-23 include (in no particular order):

1. John Wall (last season played: 2020-21)
2. Kawhi Leonard (last season played: 2020-21)
3. Zion Williamson (last season played: 2020-21)
4. Jamal Murray (last season played: 2020-21)
5. Ben Simmons (last season played: 2020-21)
6. Kendrick Nunn (last season played: 2020-21)
7. T.J. Warren (last season played: 2019-20; only 4 games played in 2020-21)

Absent data from the 2021-22 season, we'll pull in their data from their last season played. Since this is such a small number of players, we'll create the dataframe manually.

```
# add in data for our missing players
Rk <- c(606, 607, 608, 609, 610, 611, 612)
Player <- c("John Wall", "Kawhi Leonard", "Zion Williamson",
    "Jamal Murray", "Ben Simmons", "Kendrick Nunn", "T.J. Warren")
PTS <- c(20.6, 24.8, 27, 21.2, 14.3, 14.6, 19.8)
TRB <- c(3.2, 6.5, 7.2, 4, 7.2, 3.2, 4.2)
AST <- c(6.9, 5.2, 3.7, 4.8, 6.9, 2.6, 1.5)
STL <- c(1.1, 1.6, 0.9, 1.3, 1.6, 0.9, 1.2)
BLK <- c(0.8, 0.4, 0.6, 0.3, 0.6, 0.3, 0.5)
eFG <- c(0.458, 0.568, 0.616, 0.559, 0.56, 0.578, 0.581)

# create dataframe from missing data
missing_players <- data.frame(Rk, Player, PTS, TRB, AST, STL,
    BLK, eFG)

# add missing players as rows to full dataframe
player_stats_final <- rbind(player_stats_final, missing_players)
```

We now have our seven missing players added in to the dataset.

Add rookies to dataset

Rookies are a little more challenging, as we don't have any NBA stat lines for them for previous years. However, we'll want to make sure they're counted in the team's depth, as top rookies can very possibly have impact as solid role players. To make sure they're accounted for in some way, we'll use bleacher report's NBA comparisons for the upcoming rookies, and use the rookie statline for their comparison player. It's rough, but it gives us some way to account for them.

1. Paolo Banchero (NBA comp: Carmelo Anthony)
2. Malaki Branham (NBA comp: Khris Middleton)
3. Dyson Daniels (NBA comp: Marcus Smart)
4. Johnny Davis (NBA comp: Josh Hart)
5. Jalen Duren (NBA comp: Derrick Favors)
6. AJ Griffin (NBA comp: Saddiq Bey)
7. Jaden Ivey (NBA comp: Victor Oladipo)
8. Benedict Mathurin (NBA comp: Tim Hardaway Jr.)
9. Keegan Murray (NBA comp: T.J. Warren)
10. Shaedon Sharpe (NBA comp: Zach LaVine)
11. Jabari Smith (NBA comp: Jaren Jackson Jr.)
12. Jeremy Sochan (NBA comp: Franz Wagner)
13. Mark Williams (NBA comp: Clint Capela)

```
# create data for rookie stat lines
Rk <- c(613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623,
    624, 625)
Player <- c("Paolo Banchero", "Malaki Branham", "Dyson Daniels",
    "Johnny Davis", "Jalen Duren", "AJ Griffin", "Jaden Ivey",
    "Benedict Mathurin", "Keegan Murray", "Shaedon Sharpe", "Jabari Smith",
    "Jeremy Sochan", "Mark Williams")
PTS <- c(21, 6.1, 7.8, 7.9, 6.8, 12.2, 13.8, 10.2, 6.1, 10.1,
    13.8, 15.2, 2.7)
TRB <- c(6.1, 1.9, 3.3, 4.2, 5.3, 4.5, 4.1, 1.5, 2.1, 2.8, 4.7,
    4.5, 3)
AST <- c(2.8, 1, 3.1, 1.3, 0.5, 1.4, 4.1, 0.8, 0.6, 3.6, 1.1,
    2.9, 0.2)
STL <- c(1.2, 0.6, 1.5, 0.7, 0.4, 0.7, 1.6, 0.5, 0.5, 0.7, 0.9,
    0.9, 0.1)
BLK <- c(0.5, 0.1, 0.3, 0.3, 0.9, 0.2, 0.5, 0.1, 0.2, 0.1, 1.4,
    0.4, 0.8)
eFG <- c(0.449, 0.489, 0.462, 0.573, 0.517, 0.53, 0.458, 0.523,
    0.54, 0.465, 0.549, 0.517, 0.483)

# create rookie dataframe
rookies <- data.frame(Rk, Player, PTS, TRB, AST, STL, BLK, eFG)

# bind to full dataframe
player_stats_final <- rbind(player_stats_final, rookies)
```

Verify data

We should now have 486 total players, with the 20 missing/rookie players added at the end.

```r
# check the final dataframe to verify the data was added
# correctly
count(player_stats_final)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   486
```

```r
player_stats_final %>%
    arrange(desc(Rk)) %>%
    head(20)
```

```
## # A tibble: 20 x 8
##       Rk Player              PTS   TRB   AST   STL   BLK   eFG
##    <dbl> <chr>             <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1   625 Mark Williams       2.7   3     0.2   0.1   0.8 0.483
##  2   624 Jeremy Sochan      15.2   4.5   2.9   0.9   0.4 0.517
##  3   623 Jabari Smith       13.8   4.7   1.1   0.9   1.4 0.549
##  4   622 Shaedon Sharpe     10.1   2.8   3.6   0.7   0.1 0.465
##  5   621 Keegan Murray       6.1   2.1   0.6   0.5   0.2 0.54
##  6   620 Benedict Mathurin  10.2   1.5   0.8   0.5   0.1 0.523
##  7   619 Jaden Ivey         13.8   4.1   4.1   1.6   0.5 0.458
##  8   618 AJ Griffin         12.2   4.5   1.4   0.7   0.2 0.53
##  9   617 Jalen Duren         6.8   5.3   0.5   0.4   0.9 0.517
## 10   616 Johnny Davis        7.9   4.2   1.3   0.7   0.3 0.573
## 11   615 Dyson Daniels       7.8   3.3   3.1   1.5   0.3 0.462
## 12   614 Malaki Branham      6.1   1.9   1     0.6   0.1 0.489
## 13   613 Paolo Banchero     21     6.1   2.8   1.2   0.5 0.449
## 14   612 T.J. Warren        19.8   4.2   1.5   1.2   0.5 0.581
## 15   611 Kendrick Nunn      14.6   3.2   2.6   0.9   0.3 0.578
## 16   610 Ben Simmons        14.3   7.2   6.9   1.6   0.6 0.56
## 17   609 Jamal Murray       21.2   4     4.8   1.3   0.3 0.559
## 18   608 Zion Williamson    27     7.2   3.7   0.9   0.6 0.616
## 19   607 Kawhi Leonard      24.8   6.5   5.2   1.6   0.4 0.568
## 20   606 John Wall          20.6   3.2   6.9   1.1   0.8 0.458
```

Clustering: First Pass

We'll use two types of clustering and compare the results, Hierarchical and K-Means. For the first pass at this analysis, we'll use a K = 5 to create five clusters of players, following qualitative bucketing based on anecdoctal NBA knowledge. NBA players can be sorted in to tiers based on ability and star power, so to create a set number of clusters, we'll use this justification for K = 5 based on qualitative NBA player tiers:

1. Superstar Players (Giannis, Steph, etc.)
2. All Star Players (Donovan Mitchell, Bradley Beal, etc.)
3. Sub-All Stars (OG Anunoby, Malcolm Brogdon, etc.)
4. Roation Players (George Hill, Cedi Osman)
5. Garbage Time Players (any player whos only playing minutes when a game has already been decided, but is still regularly on the roster)

Scale the dataframe

We'll scale our dataframe variables before starting to cluster, since we're using points, assists, and rebounds as our clustering variables, which are incomparable to each other/different units.

```
# select only numeric values of dataframe, and scale all
scaled_stats <- player_stats_final %>%
    select(PTS, TRB, AST) %>%
    scale()
```

Hierarchical Clustering

```
# create dist matrix
hclust_dist <- dist(scaled_stats)

# create clusters
hc_stats <- hclust(hclust_dist, method = "complete")

# retrieve clusters
hclust_assignments <- cutree(hc_stats, k = 5)

# assign clusters back to original df
player_stats_final_hclus <- mutate(player_stats_final, hclus = hclust_assignments)
```

K means clustering

```
# create cluster model
kmeans_model <- kmeans(scaled_stats, centers = 5)

# add clusters to original dataset
player_stats_final_kclus <- mutate(player_stats_final, kclus = kmeans_model$cluster)
```

Results

Overall, neither of these results got us quite what we wanted. The goal was to try and identify groups of "best" to "worst" players; the hierarchical results seem to be more sensitive to high-end outliers, in that there are only 10 total players in 2/5 clusters. Whereas the k-means results are little more distributed.

It seems like the data being used to create the clusters is throwing the groupings off - while both methods do a decent job of creating groups based on PTS, the AST and TRB seem to be creating separation among top players. Thinking about this data, this would make sense given that there are different archetypes of players.

For example, neither method results in Giannis Antetokounmpo and Stephen Curry being in the same cluster. Theoretically going by "best" player, these two should be a given in the "top" cluster, as they are undisputed two of the best players in the league. However, given their different player archetypes (smaller PG versus larger PF), this creates disparities. So while they're both likely to average similar PTS values, their AST and TRB values will naturally be quite different, and result in different cluster placements.

Ultimately, although how "good" a player is is subjective, to an extent, there are some obvious flaws as noted above. This isn't necessarily wrong, but for the context of clustering players by generally how good they are, a subjective knowledge of the data tells you this is not accurate.

```
# view summary of hierarchical cluster groups
player_stats_final_hclus %>%
    group_by(hclus) %>%
    summarize(num_players = n(), mean_PTS = mean(PTS), mean_AST = mean(AST),
        mean_TRB = mean(TRB))
```

```
## # A tibble: 5 x 5
##   hclus num_players mean_PTS mean_AST mean_TRB
##   <int>       <int>    <dbl>    <dbl>    <dbl>
## 1     1         362     7.56     1.38     3.27
## 2     2          41    15.9      2.79     9.08
## 3     3           6    29.4      6.53    10.3
## 4     4          73    16.7      5.24     4.26
## 5     5           4    21.6     10        6.02
```

```
# view summary of k-means cluster groups
player_stats_final_kclus %>%
    group_by(kclus) %>%
    summarize(num_players = n(), mean_PTS = mean(PTS), mean_AST = mean(AST),
        mean_TRB = mean(TRB))
```

```
## # A tibble: 5 x 5
##   kclus num_players mean_PTS mean_AST mean_TRB
##   <int>       <int>    <dbl>    <dbl>    <dbl>
## 1     1         169     4.77     1.01     2.06
## 2     2          94    12.7      3.29     3.49
## 3     3         133     8.69     1.30     4.63
## 4     4          39    15.3      2.61     9.30
## 5     5          51    21.8      6.31     5.82
```

New Metrics Approach: Gathering the Data

Before writing off the clustering approach, we'll trying using some different metrics for the clustering. Looking through basketball-reference.com, this time we'll try using some advanced stats that are more generalized for win contribution and player. There are quite a few of these, but to start we'll use some generic ones that aren't offense/defense-specific (as that might lead us to the same clustering issue that AST and TRB showed). Instead, we'll use Win Share (WS) and Value over Replacement Player (VORP). Data once again from basketball-reference.com 2021-22 player advanced stats.

Importing and cleaning the data

The data for advanced stats is contained in a different table on basketball reference, so we'll repeat the import and cleaning steps done for the previous dataset, but with less filler explanation (the actual set of players and structure of the data is the same, just a different set of stats in the variable columns).

We end up with a cleaned, de-duped dataset, limited to our player name, Win Share (WS), and Value over Replacement Player (VORP). This results in 466 observations, which checks out against our original dataset (pre-addition of rookies and missing players).

```r
# import new dataset
advanced_stats_raw <- read_csv("player_20212022_advanced_stats.csv")

# remove duplicate rows, same as before
advanced_stats_nodupes <- advanced_stats_raw %>%
    distinct(Rk, Player, .keep_all = TRUE)

# keep only players with 15 games or 5 starts the advanced
# stats dataset does not have 'Games Started' like the
# prevoius dataset, so we'll use a semi-join on the
# previous dataset to filter for only those players that
# were kept before, based on games player and games started
advanced_stats_cleaned <- advanced_stats_nodupes %>%
    semi_join(player_stats_final, by = "Player")

# keep only variables of interest
advanced_stats_cleaned <- advanced_stats_cleaned %>%
    select(Player, WS, VORP)

# count to make sure
count(advanced_stats_cleaned)
```

Adding missing players and rookies

Same process as before, but with the new WS and VORP stats. We'll do this in one step this time since the process/logic is the same.

```r
# create data vectors
Player <- c("John Wall", "Kawhi Leonard", "Zion Williamson",
    "Jamal Murray", "Ben Simmons", "Kendrick Nunn", "T.J. Warren",
    "Paolo Banchero", "Malaki Branham", "Dyson Daniels", "Johnny Davis",
    "Jalen Duren", "AJ Griffin", "Jaden Ivey", "Benedict Mathurin",
    "Keegan Murray", "Shaedon Sharpe", "Jabari Smith", "Jeremy Sochan",
    "Mark Williams")
WS <- c(-0.2, 8.8, 8.7, 4.6, 6, 2.9, 0.1, 6.1, 0.9, 2.9, 3.4,
    2.9, 3.3, 1.3, 3.1, 1.1, -0.7, 3.3, 4, 0)
VORP <- c(0.4, 4.2, 4, 1.7, 2.3, 0.6, -0.1, 1.6, 0, 0.7, 0.7,
    -0.5, 0.7, 0.3, 0.1, 0.2, -1.3, 0.7, 0.8, -0.1)

# create dataframe for missing data
missing_advanced_stats <- data.frame(Player, WS, VORP)

# bind data to full dataframe
advanced_stats_final <- rbind(advanced_stats_cleaned, missing_advanced_stats)

# check count to make sure data was added succcesfully
count(advanced_stats_final)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   486
```

Clustering: Advanced Stats

Scale the data

```
# scale dataframe
scaled_advanced <- advanced_stats_final %>%
    select(WS, VORP) %>%
    scale()
```

Hierarchical Clustering: Advanced Stats

```
# create dist matrix
hclust_adv <- dist(scaled_advanced)

# create clusters
hc_adv <- hclust(hclust_adv, method = "complete")

# retrieve clusters
hclust_assignments_adv <- cutree(hc_adv, k = 5)

# assign clusters back to original df
advanced_stats_final_hclus <- mutate(advanced_stats_final, hclus = hclust_assignments_adv)
```

K-means Clustering: Advanced Stats

```
# k-means clustering for the advanced stats data

# create cluster model
set.seed(123)
kmeans_model_advanced <- kmeans(scaled_advanced, centers = 5)

# add clusters to original dataset
advanced_stats_final_kclus <- mutate(advanced_stats_final, kclus = kmeans_model_advanced$cluster)
```

Results

Clustering by these metrics seems to be more accurate, as we see a general trend up in both metric, without categorical variation. It does still appear that hierarchical clustering created a more exclusive top-tier than k-means.

```
# view summary of hierarchical cluster groups for advanced
# stats clustering
advanced_stats_final_hclus %>%
    group_by(hclus) %>%
    summarize(num_players = n(), mean_WS = mean(WS), mean_VORP = mean(VORP))
```

```
## # A tibble: 5 x 4
##   hclus num_players mean_WS mean_VORP
##   <int>       <int>   <dbl>     <dbl>
## 1     1         194   2.42      0.431
## 2     2          85   5.41      1.61
## 3     3          32   8.07      3.71
## 4     4         172   0.496    -0.226
## 5     5           3  13.4       7.9
```

```
# view summary of k-means cluster groups for advanced stats
# clustering
advanced_stats_final_kclus %>%
    group_by(kclus) %>%
    summarize(num_players = n(), mean_WS = mean(WS), mean_VORP = mean(VORP))
```

```
## # A tibble: 5 x 4
##   kclus num_players mean_WS mean_VORP
##   <int>       <int>   <dbl>     <dbl>
## 1     1         145   2.53      0.429
## 2     2          16   9.79      5.13
## 3     3          79   4.76      1.35
## 4     4         205   0.611    -0.166
## 5     5          41   7.07      2.61
```

Calculate Team Depth

Assigne player value scores

First we'll calculate player "value" based on their cluster. We'll qualitatively assign a "value score" based on the cluster each player is in, with higher value score for clusters with better WS/VORP metrics. We'll then use those value scores to total up score by team, to assess team depth by cluster of how "good" a player is.

Values scores will be assigned as below, and are loosely based on the average WS metric for each cluster. We'll give the top-tier players in the h-clusters more weight since those clusters are more exclusive.

H-Clusters:

- Cluster 1: 3 pts
- Cluster 2: 5 pts
- Cluster 3: 10 pts
- Cluster 4: 1 pt
- Cluster 5: 15 pts

K-Clusters:

- Cluster 1: 3 pts
- Cluster 2: 10 pts
- Cluster 3: 5 pts
- Cluster 4: 1 pt
- Cluster 5: 7 pts

```
# create unified datafrom with players, both cluster
# values, and original advanced stats
player_values <- data.frame(advanced_stats_final$Player, advanced_stats_final$WS,
    advanced_stats_final$VORP, advanced_stats_final_hclus$hclus,
    advanced_stats_final_kclus$kclus)

# rename columns to clean up dataframe
player_values <- player_values %>%
    rename(Player = advanced_stats_final.Player, WS = advanced_stats_final.WS,
        VORP = advanced_stats_final.VORP, hclus = advanced_stats_final_hclus.hclus,
        kclus = advanced_stats_final_kclus.kclus)

# initialize value scores based on cluster, as oulined
# above
index <- c(1, 2, 3, 4, 5)
h_values <- c(3, 5, 10, 1, 15)
k_values <- c(3, 10, 5, 1, 7)

# assign value scores in two new columns added to the
# dataframe
player_values$h_value <- NA
player_values$k_value <- NA
player_values$h_value <- h_values[match(player_values$hclus,
    index)]
player_values$k_value <- k_values[match(player_values$kclus,
    index)]
```

Bring in current season rosters

Since the data used to assign clusters is from the 2021-2022 season, we'll need to bring in one more data file to pull in the current teams for each player for the 2022-2023 season. Roster data obtained from realgm.com 2022-2023 NBA Player list.

```
# import current rosters the data has been paired down in
# the CSV to only include player name, team, and position
# we also need to set the encoding to UTF-8 to account for
# foreign characters, as many players have
# serbian/slovenian characters in their names
current_rosters <- read.csv("2022-23_player_roster.csv", encoding = "UTF-8")

player_values <- player_values %>%
    left_join(current_rosters, by = c("Player"))

player_values %>%
    filter(is.na(Team))
```

```
##                    Player   WS VORP hclus kclus h_value k_value Position Team
## 1        LaMarcus Aldridge  3.1  0.7     1     1       3       3     <NA> <NA>
## 2          Justin Anderson  0.4 -0.1     4     4       1       1     <NA> <NA>
## 3          Carmelo Anthony  3.6  0.7     1     1       3       3     <NA> <NA>
## 4             Trevor Ariza  0.2 -0.2     4     4       1       1     <NA> <NA>
## 5            D.J. Augustin  0.8 -0.4     4     4       1       1     <NA> <NA>
```

```
## 6           Charles Bassey  0.8  0.2     1     4       3       1     <NA> <NA>
## 7            Kent Bazemore  0.1 -0.3     4     4       1       1     <NA> <NA>
## 8          DeAndre' Bembry  2.2  0.3     1     1       3       3     <NA> <NA>
## 9          Nemanja Bjelica  3.0  1.0     1     1       3       3     <NA> <NA>
## 10            Eric Bledsoe  1.3  0.4     1     1       3       3     <NA> <NA>
## 11          Keljin Blevins -0.5 -0.6     4     4       1       1     <NA> <NA>
## 12      Brandon Boston Jr.  0.4 -0.3     4     4       1       1     <NA> <NA>
## 13           Avery Bradley  1.2 -0.6     4     4       1       1     <NA> <NA>
## 14        Ignas Brazdeikis  0.5 -0.3     4     4       1       1     <NA> <NA>
## 15           Miles Bridges  7.2  2.5     2     5       5       7     <NA> <NA>
## 16           Armoni Brooks  0.2 -0.4     4     4       1       1     <NA> <NA>
## 17       Charlie Brown Jr.  0.0 -0.1     4     4       1       1     <NA> <NA>
## 18          Sterling Brown  0.8 -0.1     4     4       1       1     <NA> <NA>
## 19              Trey Burke  0.0 -0.3     4     4       1       1     <NA> <NA>
## 20            Jared Butler  0.4  0.0     4     4       1       1     <NA> <NA>
## 21     Willie Cauley-Stein  0.2  0.0     4     4       1       1     <NA> <NA>
## 22           Chris Chiozza  0.0 -0.4     4     4       1       1     <NA> <NA>
## 23         Marquese Chriss  0.8 -0.2     4     4       1       1     <NA> <NA>
## 24              Gary Clark  0.7  0.2     4     4       1       1     <NA> <NA>
## 25              Tyler Cook  0.5 -0.1     4     4       1       1     <NA> <NA>
## 26        DeMarcus Cousins  1.6  0.1     1     4       3       1     <NA> <NA>
## 27                Ed Davis  0.5  0.0     4     4       1       1     <NA> <NA>
## 28              PJ Dozier  0.3  0.0     4     4       1       1     <NA> <NA>
## 29              CJ Elleby  0.4 -1.0     4     4       1       1     <NA> <NA>
## 30         Wayne Ellington  1.0  0.1     1     4       3       1     <NA> <NA>
## 31          Derrick Favors  1.4  0.0     1     4       3       1     <NA> <NA>
## 32            Enes Freedom  1.4  0.1     1     4       3       1     <NA> <NA>
## 33         Brandon Goodwin  0.7  0.1     4     4       1       1     <NA> <NA>
## 34                Kyle Guy  0.1 -0.1     4     4       1       1     <NA> <NA>
## 35        Maurice Harkless  0.5 -0.3     4     4       1       1     <NA> <NA>
## 36            Jaylen Hoard  0.6  0.1     4     4       1       1     <NA> <NA>
## 37             Rodney Hood  0.7 -0.2     4     4       1       1     <NA> <NA>
## 38           Dwight Howard  3.5  0.4     1     1       3       3     <NA> <NA>
## 39           Markus Howard  0.3  0.0     4     4       1       1     <NA> <NA>
## 40           Elijah Hughes -0.4 -0.5     4     4       1       1     <NA> <NA>
## 41           Frank Jackson  0.7 -0.4     4     4       1       1     <NA> <NA>
## 42            Josh Jackson  0.0 -0.5     4     4       1       1     <NA> <NA>
## 43           Alize Johnson  0.1 -0.1     4     4       1       1     <NA> <NA>
## 44         Stanley Johnson  1.8 -0.2     4     4       1       1     <NA> <NA>
## 45             Jeremy Lamb  1.2  0.3     1     4       3       1     <NA> <NA>
## 46             Jake Layman  0.1 -0.1     4     4       1       1     <NA> <NA>
## 47               Saben Lee  0.9  0.2     1     4       3       1     <NA> <NA>
## 48 Timothé Luwawu-Cabarrot  0.9 -0.2     4     4       1       1     <NA> <NA>
## 49            Kelan Martin  0.0 -0.2     4     4       1       1     <NA> <NA>
## 50             Skylar Mays  0.3  0.0     4     4       1       1     <NA> <NA>
## 51        Alfonzo McKinnie -0.1 -0.4     4     4       1       1     <NA> <NA>
## 52            Ben McLemore  0.8 -0.4     4     4       1       1     <NA> <NA>
## 53            Paul Millsap  0.5 -0.1     4     4       1       1     <NA> <NA>
## 54           Mychal Mulder -0.2 -0.3     4     4       1       1     <NA> <NA>
## 55             David Nwaba  1.3  0.2     1     4       3       1     <NA> <NA>
## 56            Semi Ojeleye  0.2 -0.4     4     4       1       1     <NA> <NA>
## 57                Miye Oni  0.0 -0.1     4     4       1       1     <NA> <NA>
## 58            Kevin Pangos  0.1 -0.2     4     4       1       1     <NA> <NA>
## 59           Elfrid Payton -0.2 -0.2     4     4       1       1     <NA> <NA>
```

```
## 60          Jahmi'us Ramsey -0.1 -0.2     4     4       1       1     <NA> <NA>
## 61          Justin Robinson -0.3 -0.4     4     4       1       1     <NA> <NA>
## 62              Rajon Rondo  0.7  0.1     4     4       1       1     <NA> <NA>
## 63        Tomáš Satoranský  1.0 -0.4     4     4       1       1     <NA> <NA>
## 64              Jay Scrubb  0.0 -0.1     4     4       1       1     <NA> <NA>
## 65           Javonte Smart -0.2 -0.2     4     4       1       1     <NA> <NA>
## 66              Tony Snell  0.4 -0.5     4     4       1       1     <NA> <NA>
## 67        Lance Stephenson  0.8 -0.2     4     4       1       1     <NA> <NA>
## 68            Keifer Sykes -0.5 -0.7     4     4       1       1     <NA> <NA>
## 69           Isaiah Thomas  0.2 -0.2     4     4       1       1     <NA> <NA>
## 70              Matt Thomas  0.6 -0.1     4     4       1       1     <NA> <NA>
## 71        Tristan Thompson  1.6 -0.3     4     4       1       1     <NA> <NA>
## 72          Killian Tillie  0.6 -0.1     4     4       1       1     <NA> <NA>
## 73        Denzel Valentine  0.2  0.0     4     4       1       1     <NA> <NA>
## 74            Kemba Walker  1.8  0.6     1     1       3       3     <NA> <NA>
## 75          Brad Wanamaker  0.0 -0.3     4     4       1       1     <NA> <NA>
## 76        Hassan Whiteside  5.8  1.2     2     3       5       5     <NA> <NA>
## 77            Joe Wieskamp  0.1 -0.1     4     4       1       1     <NA> <NA>
## 78       Lindell Wigginton  0.2 -0.2     4     4       1       1     <NA> <NA>
## 79        Brandon Williams -0.5 -0.6     4     4       1       1     <NA> <NA>
## 80            Lou Williams  0.6 -0.3     4     4       1       1     <NA> <NA>
## 81              Cody Zeller  1.1  0.0     1     4       3       1     <NA> <NA>
```

It looks like the data joined in well for the most part, but we have a fair amount of players who don't have any team data for the current season. This is to be expected, as there are many players who transition out of the league, or are free agents/not signed to a team to start the season. Initially, there were some active players who did not match up, but this data was then cleaned externally in the CSV to account for mismatches in name abbreviations/punctuations/etc. All players not actively on a team's roster to start the season will not be counted towards depth calculations.

Depth Calculations

Validate Team Rosters

First we'll run a quick check on our data to ensure that, between filtering out players with few games, and unsigned players, each team has at least 9 players left in our data set. We'll calculate depth based on the top 9 players per team, as those are usually the players that receive consistent playing time during the regular season

```
player_values %>%
    group_by(Team) %>%
    summarize(num_players = n())
```

```
## # A tibble: 31 x 2
##    Team                  num_players
##    <chr>                       <int>
##  1 Atlanta Hawks                  14
##  2 Boston Celtics                 12
##  3 Brooklyn Nets                  15
##  4 Charlotte Hornets              15
##  5 Chicago Bulls                  14
```

```
##  6 Cleveland Cavaliers            14
##  7 Dallas Mavericks               14
##  8 Denver Nuggets                 12
##  9 Detroit Pistons                15
## 10 Golden State Warriors          12
## # ... with 21 more rows
```

Team counts look good, we have at least 11 players per team remaining in our dataset.

Keep Top 9 Players Per Team

As mentioned, we'll calculate depth based on player values for the top 9 players on each team. Since each team has between 11-15 players in our dataset, we don't want to skew the calculations by comparing different numbers of players on each team. We'll do this for both the player values based on hierarchical clustering and k-means clustering.

```
# top 9 players per team by h_value
top_players_h <- player_values %>%
    group_by(Team) %>%
    slice_max(order_by = h_value, n = 9, with_ties = FALSE)

# top 9 players per team by k_value
top_players_k <- player_values %>%
    group_by(Team) %>%
    slice_max(order_by = k_value, n = 9, with_ties = FALSE)

# validate
top_players_h %>%
    group_by(Team) %>%
    summarize(num_players = n())
```

```
## # A tibble: 31 x 2
##    Team                  num_players
##    <chr>                       <int>
##  1 Atlanta Hawks                   9
##  2 Boston Celtics                  9
##  3 Brooklyn Nets                   9
##  4 Charlotte Hornets               9
##  5 Chicago Bulls                   9
##  6 Cleveland Cavaliers             9
##  7 Dallas Mavericks                9
##  8 Denver Nuggets                  9
##  9 Detroit Pistons                 9
## 10 Golden State Warriors           9
## # ... with 21 more rows
```

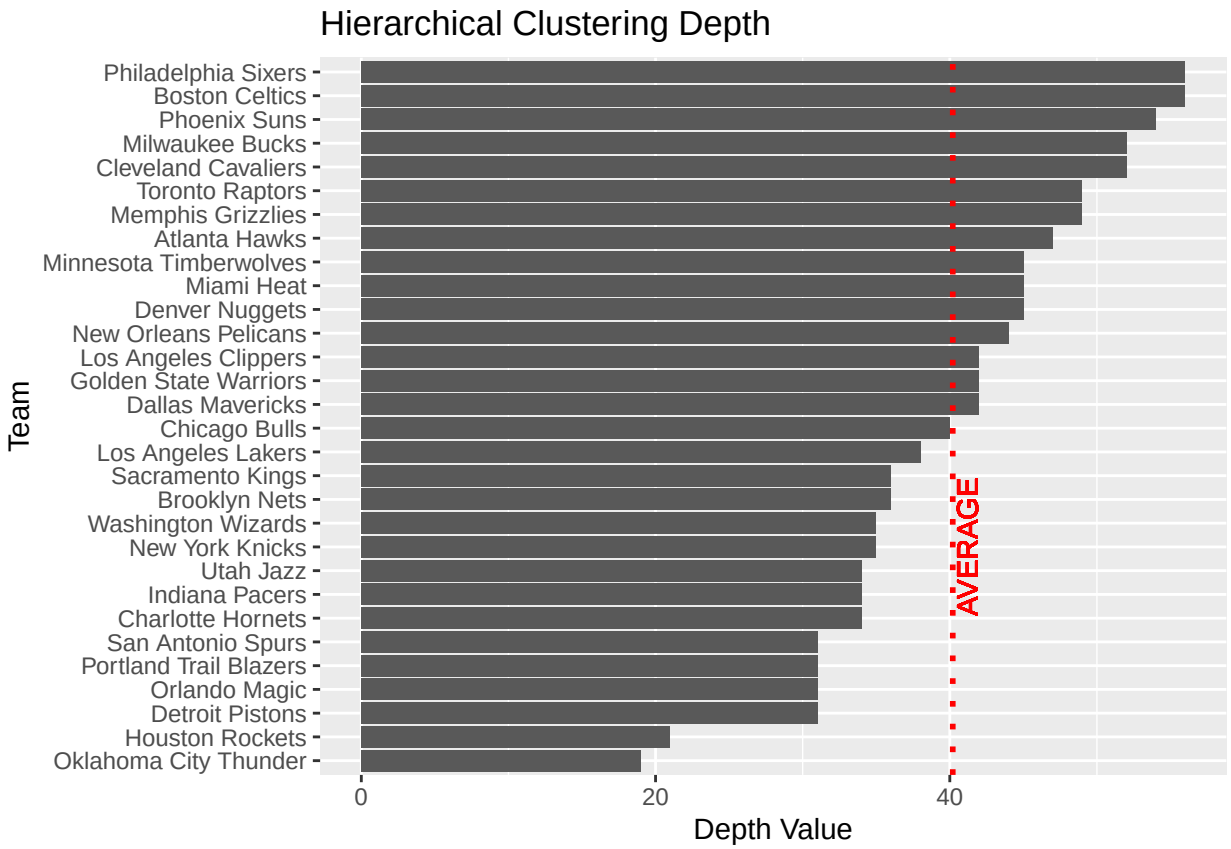
```
top_players_k %>%
    group_by(Team) %>%
    summarize(num_players = n())
```

```
## # A tibble: 31 x 2
##    Team                  num_players
```

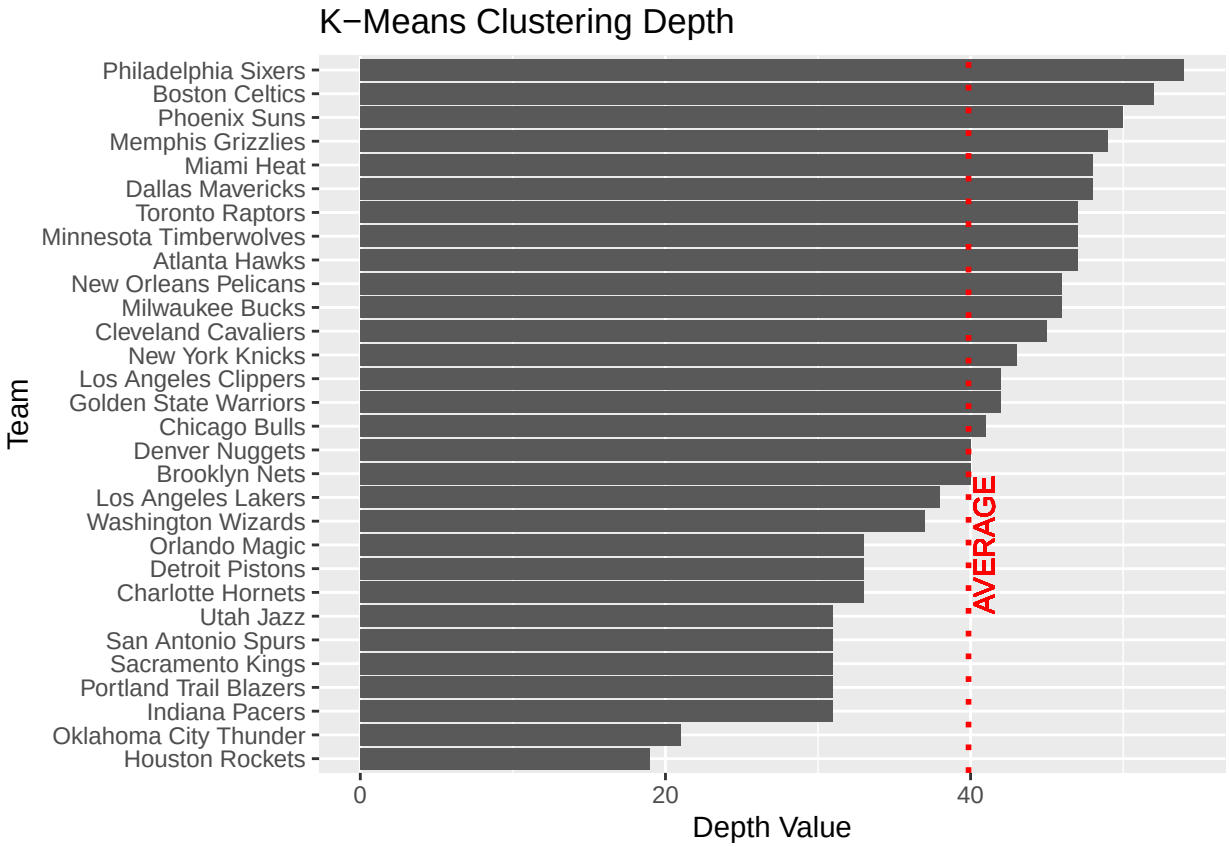
```
##    <chr>                      <int>
##  1 Atlanta Hawks                  9
##  2 Boston Celtics                 9
##  3 Brooklyn Nets                  9
##  4 Charlotte Hornets              9
##  5 Chicago Bulls                  9
##  6 Cleveland Cavaliers            9
##  7 Dallas Mavericks               9
##  8 Denver Nuggets                 9
##  9 Detroit Pistons                9
## 10 Golden State Warriors          9
## # ... with 21 more rows
```

Initial Depth Calculation

```
# calculate depth based on h_value
h_depth <- top_players_h %>%
    group_by(Team) %>%
    summarize(depth_value = sum(h_value)) %>%
    filter(!is.na(Team)) %>%
    arrange(desc(depth_value))
h_depth_avg <- mean(h_depth$depth_value)

# calculate depth based on k_value
k_depth <- top_players_k %>%
    group_by(Team) %>%
    summarize(depth_value = sum(k_value)) %>%
    filter(!is.na(Team)) %>%
    arrange(desc(depth_value))
k_depth_avg <- mean(k_depth$depth_value)

# print
h_depth %>%
    ggplot(aes(x = depth_value, y = reorder(Team, depth_value))) +
    geom_col() + xlab("Depth Value") + ylab("Team") + ggtitle("Hierarchical Clustering Depth") +
    geom_vline(xintercept = h_depth_avg, linetype = "dotted",
        color = "red", size = 1) + geom_text(aes(x = h_depth_avg +
    1, y = 10, label = "AVERAGE"), color = "red", angle = 90)
```




```
k_depth %>%
    ggplot(aes(x = depth_value, y = reorder(Team, depth_value))) +
    geom_col() + xlab("Depth Value") + ylab("Team") + ggtitle("K-Means Clustering Depth") +
    geom_vline(xintercept = k_depth_avg, linetype = "dotted",
        color = "red", size = 1) + geom_text(aes(x = k_depth_avg +
    1, y = 10, label = "AVERAGE"), color = "red", angle = 90)
```



Ranking Clustering Values

To get one final ranking, we'll average out the depth value from both clustering methods, and rank the teams according to that value.

```
# calculate average depth from both clustering methods
final_depth <- h_depth %>%
    left_join(k_depth, by = c("Team"))
final_depth <- final_depth %>%
    mutate(Team, h_depth = depth_value.x, k_depth = depth_value.y)


# rank teams by depth value
final_depth_ranked <- final_depth %>%
    transmute(Team, h_depth_rank = rank(desc(h_depth)), k_depth_rank = rank(desc(k_depth)))
final_depth_ranked
```

```
## # A tibble: 30 x 3
##    Team                h_depth_rank k_depth_rank
##    <chr>                      <dbl>        <dbl>
##  1 Boston Celtics               1.5          2
##  2 Philadelphia Sixers          1.5          1
##  3 Phoenix Suns                 3            3
##  4 Cleveland Cavaliers          4.5         12
##  5 Milwaukee Bucks              4.5         10.5
```

```
##  6 Memphis Grizzlies            6.5          4
##  7 Toronto Raptors              6.5          8
##  8 Atlanta Hawks                8            8
##  9 Denver Nuggets              10           17.5
## 10 Miami Heat                  10            5.5
## # ... with 20 more rows
```

Model Accuracy

Assumption

To determine model accuracy, we'll compare the rankings based on depth values to the nba.com power rankings. For the purposes of this analysis, we'll assume these power rankings are the gold standard that the clustering models should result in.

Depth Fit vs. Power Rankings

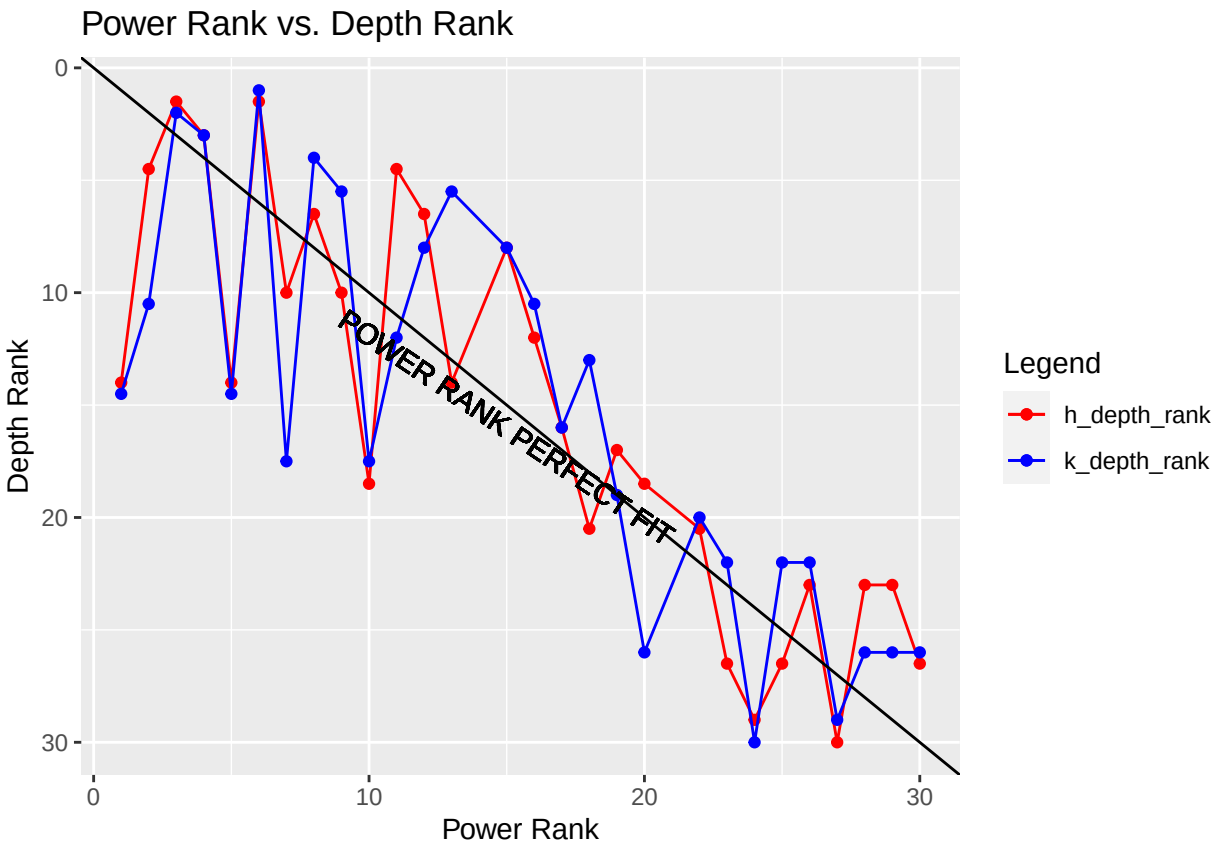
Power rankings are common in most major sports leagues, ranking teams from best to worst at points throughout the season. We'll compare the ranking developed through the depth calculations to nba.com week 1 power rankings for the 2022-23 season to see how depth matches up to the experts' rankings.

```
# create power ranking data

Team <- c("Philadelphia Sixers", "Boston Celtics", "Phoenix Suns",
    "Milwaukee Bucks", "Memphis Grizzlies", "Cleveland Cavaliers",
    "Toronto Raptors", "Atlanta Hawks", "Miami Heat", "Minnesota Timeberwolves",
    "New Orleans Pelicans", "Dallas Mavericks", "Denver Nuggets",
    "Golden State Warriors", "Los Angeles Clippers", "Chicago Bulls",
    "New York Knicks", "Los Angeles Lakers", "Brooklyn Nets",
    "Washington Wizards", "Sacramento Kings", "Charlotte Hornets",
    "Indiana Pacers", "Utah Jazz", "Detroit Pistons", "Orlando Magic",
    "Portland Trailblazers", "San Antonio Spurs", "Houston Rockets",
    "Oklahoma City Thunder")

power_rank <- c(6, 3, 4, 2, 8, 11, 12, 15, 9, 14, 16, 13, 7,
    1, 5, 17, 18, 19, 10, 22, 20, 26, 29, 28, 25, 23, 21, 30,
    24, 27)

power_rankings <- data.frame(Team, power_rank)

# join to depth rank data
rank_comparison <- power_rankings %>%
    inner_join(final_depth_ranked, by = c("Team")) %>%
    mutate(h_diff = power_rank - h_depth_rank, k_diff = power_rank -
        k_depth_rank)
rank_comparison
```

```
##                    Team power_rank h_depth_rank k_depth_rank h_diff k_diff
## 1    Philadelphia Sixers          6          1.5          1.0    4.5    5.0
## 2         Boston Celtics          3          1.5          2.0    1.5    1.0
## 3           Phoenix Suns          4          3.0          3.0    1.0    1.0
```

```
## 4         Milwaukee Bucks          2          4.5         10.5   -2.5   -8.5
## 5       Memphis Grizzlies          8          6.5          4.0    1.5    4.0
## 6     Cleveland Cavaliers         11          4.5         12.0    6.5   -1.0
## 7         Toronto Raptors         12          6.5          8.0    5.5    4.0
## 8           Atlanta Hawks         15          8.0          8.0    7.0    7.0
## 9              Miami Heat          9         10.0          5.5   -1.0    3.5
## 10  New Orleans Pelicans         16         12.0         10.5    4.0    5.5
## 11      Dallas Mavericks         13         14.0          5.5   -1.0    7.5
## 12        Denver Nuggets          7         10.0         17.5   -3.0  -10.5
## 13 Golden State Warriors          1         14.0         14.5  -13.0  -13.5
## 14  Los Angeles Clippers          5         14.0         14.5   -9.0   -9.5
## 15         Chicago Bulls         17         16.0         16.0    1.0    1.0
## 16       New York Knicks         18         20.5         13.0   -2.5    5.0
## 17    Los Angeles Lakers         19         17.0         19.0    2.0    0.0
## 18         Brooklyn Nets         10         18.5         17.5   -8.5   -7.5
## 19    Washington Wizards         22         20.5         20.0    1.5    2.0
## 20      Sacramento Kings         20         18.5         26.0    1.5   -6.0
## 21     Charlotte Hornets         26         23.0         22.0    3.0    4.0
## 22        Indiana Pacers         29         23.0         26.0    6.0    3.0
## 23             Utah Jazz         28         23.0         26.0    5.0    2.0
## 24       Detroit Pistons         25         26.5         22.0   -1.5    3.0
## 25         Orlando Magic         23         26.5         22.0   -3.5    1.0
## 26     San Antonio Spurs         30         26.5         26.0    3.5    4.0
## 27       Houston Rockets         24         29.0         30.0   -5.0   -6.0
## 28 Oklahoma City Thunder         27         30.0         29.0   -3.0   -2.0
```

```
# visualize
ggplot(rank_comparison, aes(x = power_rank)) + geom_line(aes(y = h_depth_rank,
    color = "h_depth_rank")) + geom_point(aes(y = h_depth_rank,
    color = "h_depth_rank")) + geom_line(aes(y = k_depth_rank,
    color = "k_depth_rank")) + geom_point(aes(y = k_depth_rank,
    color = "k_depth_rank")) + geom_abline(intercept = 0, slope = -1) +
    scale_y_continuous(trans = "reverse") + geom_text(aes(x = 15,
    y = 16, label = "POWER RANK PERFECT FIT"), color = "black",
    angle = 326) + xlab("Power Rank") + ylab("Depth Rank") +
    ggtitle("Power Rank vs. Depth Rank") + scale_color_manual(name = "Legend",
    values = c(h_depth_rank = "red", k_depth_rank = "blue"))
```



Variation between the two clustering methods

We'll also calculate the average difference between each model's depth value for each team, and that team's power rank, to see if, on average, one clustering method did better than the other.

```
rank_comparison %>%
    summarize(mean_h_diff = mean(abs(h_diff)), mean_k_diff = mean(abs(k_diff)))
```

```
##   mean_h_diff mean_k_diff
## 1       3.875    4.571429
```

Results indicate the the hierarchical clustering method resulted in depth rankings that aligned slightly better to the power ranking, with an average difference in ranking of 3.875, as compared to 4.571 when using k-means.

Accuracy Takeaway

Our depth rankings did an okay job of identify a general trend in good/bad teams, but there are a few significant outliers, particularly with regards to under-rating teams by depth, at the top of the power rankings. Golden State is the most egregious in terms of depth being ranked far below their power ranking, followed by the Clippers, Brooklyn, and Denver. Atlanta's depth was the significant over-rating compared to their power ranking. Overall, this model isn't accurate enough to make predictions of the singular best team, but can give a good idea of general strength of a team (that is under the assumption we take the nba.com power rankings as the gold-standard of prediction/ranking).

Top and Bottom Teams

Using our rankings calculated via hierarchical clustering (the slightly more accurate of the models), we can determine the top and bottom teams.

```
# top 5 teams
final_depth_ranked %>%
    arrange(h_depth_rank) %>%
    head(5) %>%
    select(Team)
```

```
## # A tibble: 5 x 1
##   Team
##   <chr>
## 1 Boston Celtics
## 2 Philadelphia Sixers
## 3 Phoenix Suns
## 4 Cleveland Cavaliers
## 5 Milwaukee Bucks
```

```
# bottom 5 teams
final_depth_ranked %>%
    arrange(h_depth_rank) %>%
    tail(5) %>%
    select(Team)
```

```
## # A tibble: 5 x 1
##   Team
##   <chr>
## 1 Orlando Magic
## 2 Portland Trail Blazers
## 3 San Antonio Spurs
## 4 Houston Rockets
## 5 Oklahoma City Thunder
```

Our top 5 teams based on team depth:

1. Boston Celtics
2. Philadelphia Sixers
3. Phoenix Suns
4. Cleveland Cavaliers
5. Milwaukee Bucks

Three of these teams were also in nba.com's top 5 in power rankings (Boston, Phoenix, Milwaukee), in slightly different order, and Phildelphai was just outside at #6 in the power rankings. So we can feel pretty good about those teams projected to be in the top of the league this year. Notable addition here is the Cleveland Cavaliers, who seem to have surprising depth. Qualitatively, Cleveland is expected to be a good, but not serious top 5 contender team, so labeling them as "good" isn't wrong, but NBA analysts would all agree they are at least a tier below the other teams in this top 5. However, may be a sleeper candidate for a sneaky good team, if we take depth as a meaningful marker for success.

Notably missing from this top 5 is the Golden State Warriors. As the #1 in power ranking and favorite to win the finals, the model seems to have a big miss here, and needs some tweak for future use.

Our bottom 5 teams based on team depth:

26. Orlando Magic
27. Portland Trail Blazers
28. San Antonio Spurs
29. Houston Rockets
30. Oklahoma City Thunder

Our model was a little less accurate in predicting the specific bottom five, only two (OKC and San Antonio) of our bottom 5 here actually appear in the power ranking bottom 5. However, the model was overall a little more accurate in scoring the bottom teams, as the remaining three lie just ouside the bottom five in power rankings, with Portland being the least accurate placement here (Orlando ranked 23/30, Houston ranked 24/30, and Portland ranked 21/30). In general, the model seems to be more generally aligned on the "bad" teams with the power rankings, due to fewer large variations akin to the Warriors in the top 5.