ML Final Project: Analysis

Joe Marlo, Andrew Pagtakhan, George Perrett, Bilal Waheed

January 27, 2020

INTRO AND DATA DESCRIPTION

Research Question: Are there underlying patterns of groupings between NBA team compensation vs. overall team skillsets?

The project is an unsupervised approach to discover underlying patterns or groupings between NBA compensation vs. overall team skillsets. It uses K-means, hierarchical, and model-based clustering along with other techniques and tools such as principal component analysis, standardizing, scaling, and web-scraping.

The data includes NBA statistics for over 3,000 players, 60+ seasons, and over 50 features per players Measurement Scales: Numerical totals and percentages for features such as: points, assists, rebounds, player attributes (height, weight, college attended, etc.) Time period: 1950 - 2017 (through 2016 - 2017 season). We are selecting data for the 2016 - 2017 season.

Key Feature Set: MP_pg: Minutes played per game FG_pg: Field goals made per game FGA_pg: Field goal attempts per game 3P_pg: 3-point shots made per game 3PA_pg 3-points shot attempts per game 2P_pg: 2-point shots made per game 2PA_pg: 2-point shots attempted per game FT_pg: Free throw shots made per game FTA_pg: Free throw shots attempted per game TRB_pg: Total rebounds (offensive + defensive) per game AST_pg: Total assists per game STL_pg: Total steals per game BLK_pg: Total blocks per game PTS_pg: Total points made per game

Additional Features: win_pct: team winning percentage during the regular seasong (Total wins / total games played) Player: First and last name of NBA player Team: NBA team player played on. For multiple teams, this is the team the player played on for the most minutes Position: NBA position, e.g. C = Center, PF = Power Forward, SG = Shooting Guard, PG = Point Guard Salary: Player salary (USD) RPM: Real Plus/Minus. Player's average impact in terms of net point differential per 100 offensive and defensive possessions VORP: Value Over Replacement Player. Measure to estimate each player's overall contribution to the team PER: Player Efficiency Rating. Measures player's overall contributions across different statistics.

Source: https://www.kaggle.com/drgilermo/nba-players-stats#player_data.cs

Analysis Methods

For this analysis, we applied the following modeling techniques to analysze NBA player data: * Principal Component Analysis * Hierarchical Clustering * K-Means * Model-Based clustering

DATA EXPLORATION AND TRANSFORMATION

The data cleaning was done in separate R scripts. https://github.com/joemarlo/ML-NBA Steps: * Filtered data to the 2016 - 2017 season * Assigned player to one team based on the most minutes he played for, including stats across all teams played for * Scraped player salaries and RPM data from ESPN website, using fuzzy matching on player names to join to the main dataset * Scaled/Transformed data using cube root and standardizing. For model-based clustering, various transformations such as Log + 1, square root, cube root, and Box-Cox were applied. The cube root yielded the best transformation to normalize the data. This was validated using QQ plots. We scaled the data for Hierarchical and K-Means to put all features on an equally weighted basis. For example, minutes played per game vs. blocks per game are on different ranges, so scaling these puts them on a comparable basis.

Group Member Roles

We all collaboratively contributed to the project in all aspects. Specifically, Joe contributed to scraping the player salary, additional player features and visualizations. Andrew contributed to cleaning the data, hierarchical/K-means models, and post-cluster analysis. Bilal contributed to the hierarchical/K-Means modeling and the write-up. George contributed to the model-based clustering and data transformation.

Analysis

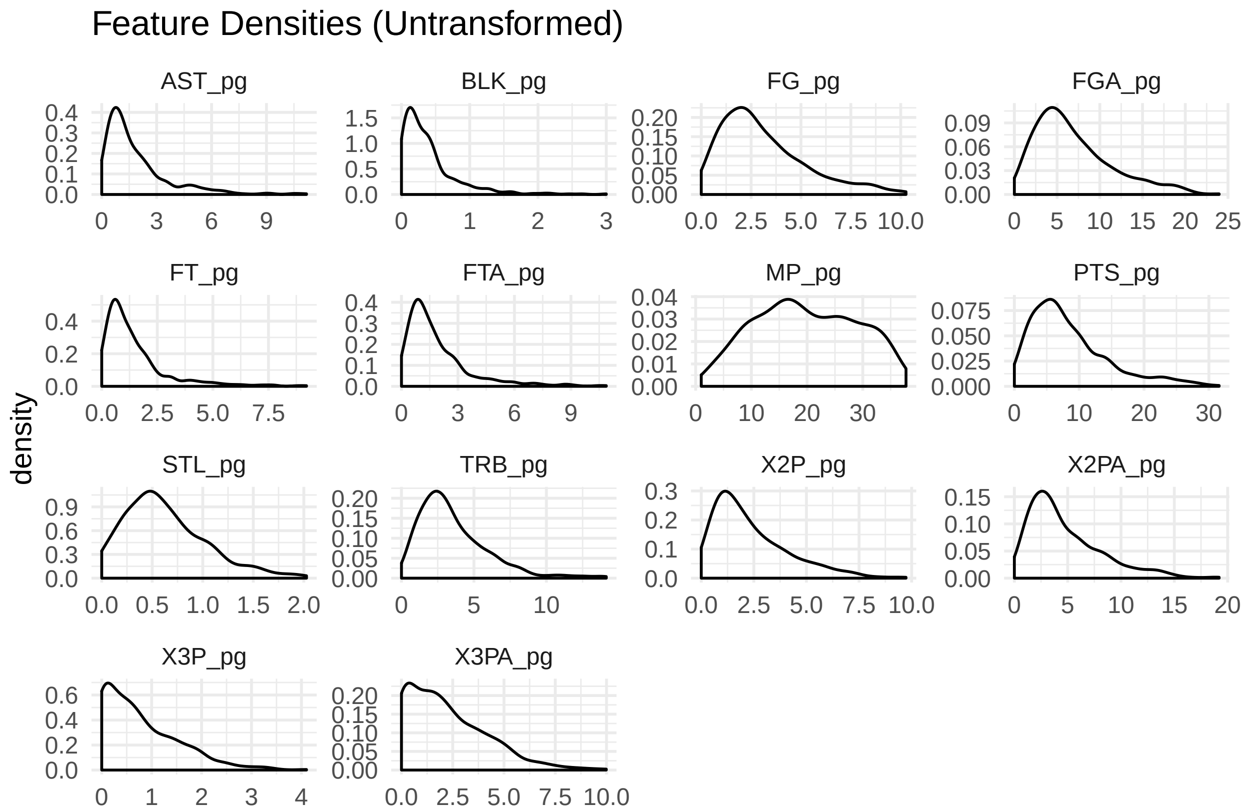
Exploratory Data Analysis

```r
library(tidyverse)
library(dendextend)
library(factoextra)
library(GGally)
library(ggfortify)
library(ggrepel)
library(gridExtra)
library(knitr)
library(mclust)
library(NbClust)
library(rgl)

# set for reproducible results
set.seed(14)

# set theme for ggplot
theme_set(theme_minimal())

# clear variables
rm(list = ls())

# set working directory
opts_knit$set(root.dir = normalizePath('..'))

# define file name for analysis
filename <- 'Data/season_stats_clean.csv'
```

We decided to use the 14 features in our analysis (on a per game basis) because they provide a good balance of offensive (e.g Pts, Ast) and defensive stats (e.g. Reb, Blk). This is more likely to provide more balanced groupings between those who are more offensive and those who are better at defense.

With these features, we will explore the distribution of each future.

```r
# load data
nba <- read.csv(filename)
# replace NA values in RPM column with 0s
nba$RPM[is.na(nba$RPM)] <- 0
```

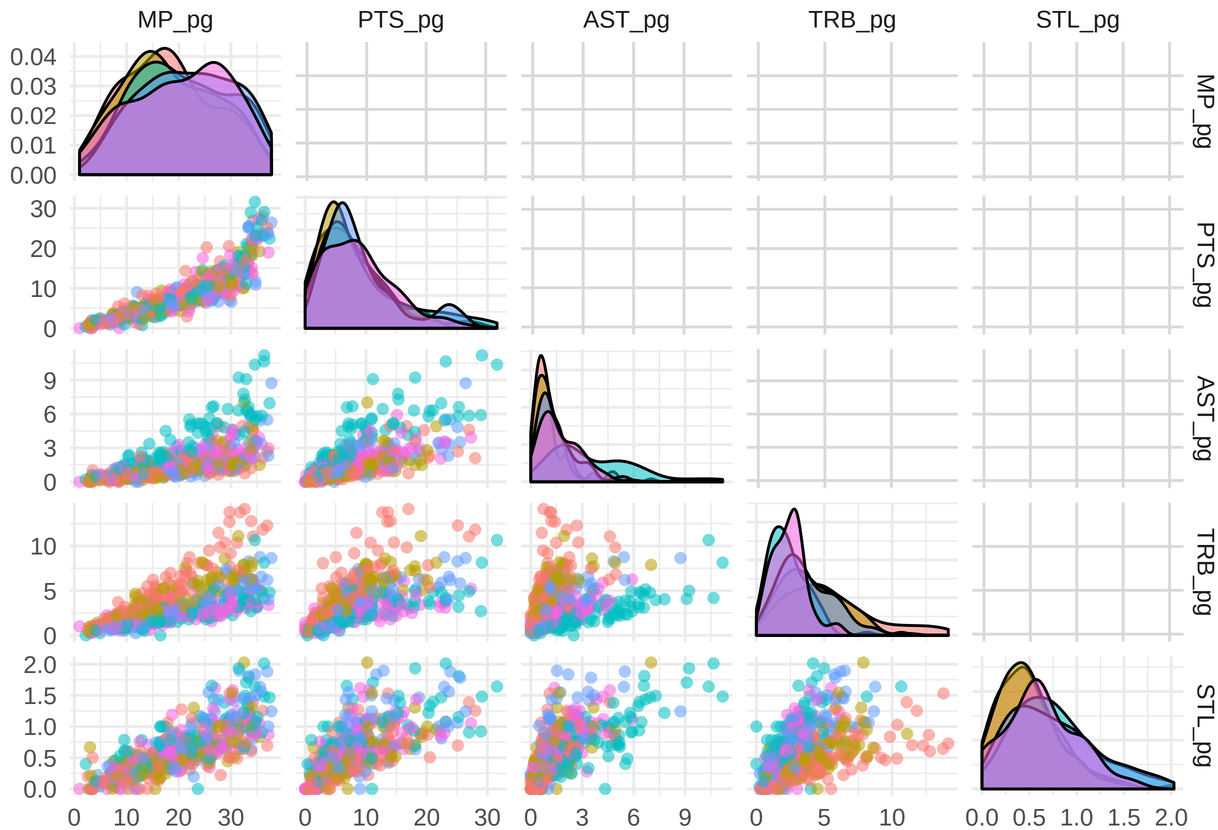


We calculate the variance to look at the spread of each feature.

```
# calculate variance
var_table <- round(apply(nba_feat, MARGIN = 2, FUN = var), 2)
var_df <- data.frame(var_table)
colnames(var_df) <- 'Variance'
kable(var_df, caption = 'Feature Variance')
```

Table 1: Feature Variance

	Variance
MP_pg	82.07
FG_pg	4.67
FGA_pg	20.49
X3P_pg	0.57
X3PA_pg	3.76
X2P_pg	3.23
X2PA_pg	11.89
FT_pg	2.04
FTA_pg	2.99
TRB_pg	5.89
AST_pg	3.11
STL_pg	0.17
BLK_pg	0.17
PTS_pg	36.72



We then scaled the data because the ranges of the data can be different. A good example is minutes and blocks per game - most players will have more minutes per game than blocks per game.

Run Principal Component Analysis

Before running any clustering algorithms, we will perform Principal Component Analysis to determine if there are any inherent groupings among players.

```
nba_feat_sc <- scale(nba_feat)

# run PCA
nba_pca <- prcomp(nba_feat_sc)
summary(nba_pca)
```

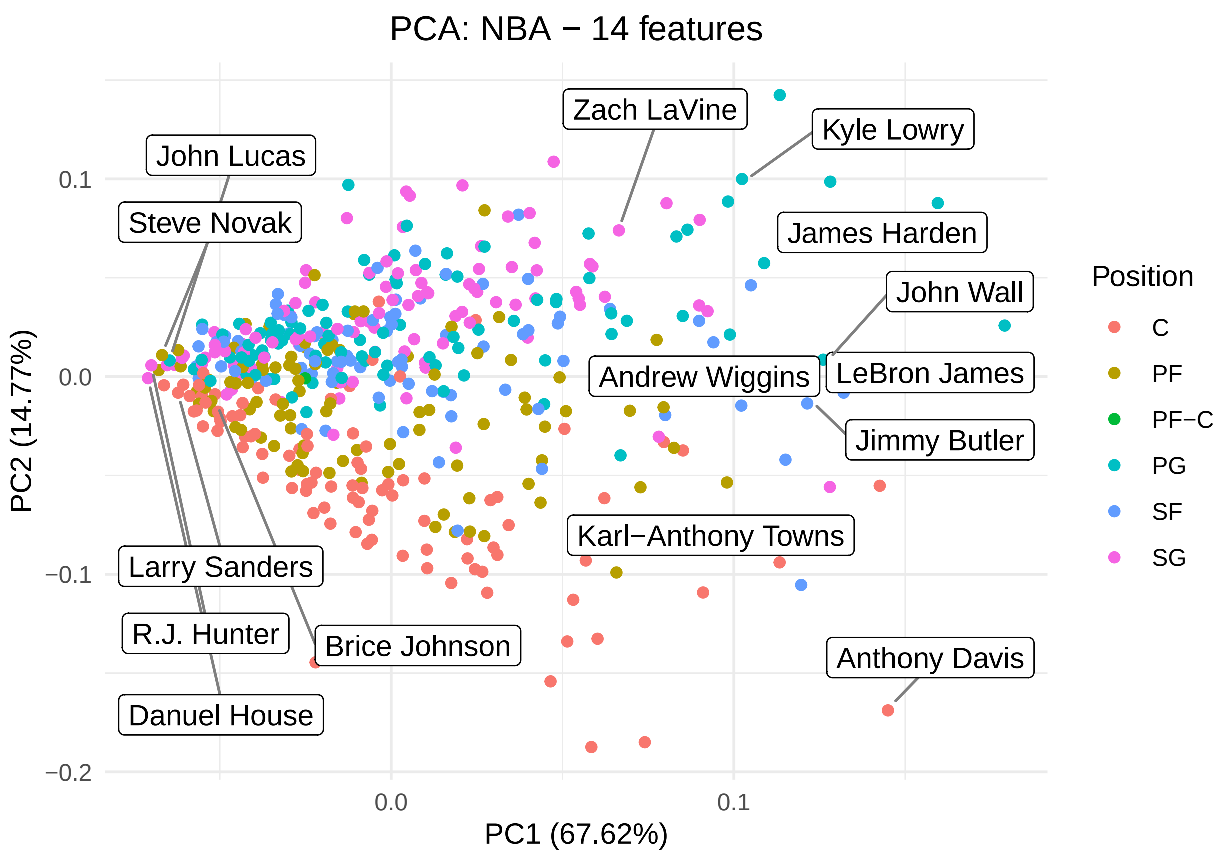
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     3.0767 1.4379 0.88672 0.80819 0.63364 0.52061
## Proportion of Variance 0.6762 0.1477 0.05616 0.04666 0.02868 0.01936
## Cumulative Proportion  0.6762 0.8239 0.88002 0.92667 0.95535 0.97471
##                            PC7     PC8     PC9    PC10    PC11      PC12
## Standard deviation     0.46651 0.29741 0.16765 0.10779 0.09065 2.849e-15
## Proportion of Variance 0.01555 0.00632 0.00201 0.00083 0.00059 0.000e+00
## Cumulative Proportion  0.99026 0.99658 0.99858 0.99941 1.00000 1.000e+00
##                             PC13      PC14
## Standard deviation     2.194e-15 1.597e-15
## Proportion of Variance 0.000e+00 0.000e+00
## Cumulative Proportion  1.000e+00 1.000e+00
```

```r
# create plot PCA data function
plot_pca <- function(object, frame = FALSE, x = 1, y = 2,
                     data, colour, title, label, leg_title) {
  # plots data in PCA space
  # object = PCA or K-Means object
  # x = which PC for x-axis (1, 2, ,3, etc..)
  # y = which PC for y-axis (1, 2, 3, etc..)
  # object: PCA or K-means object
  # data = underlying data
  p <- autoplot(nba_pca, x = x, y = y, data = nba, colour = colour, frame = frame) +
        ggtitle(title) +
        # center title
        theme(plot.title = element_text(hjust = 0.5)) +
        geom_label_repel(aes(label = label),
                        box.padding   = 0.35,
                        point.padding = 0.5,
                        segment.color = 'grey50') +
    ## This is supposed to override the autoplot legend title.
    ## Only works when plotting PCA directly. Does not work for HCL and KM objects
    labs(colour = leg_title)
        # theme_classic()
  return(p)
}
```

The first 2 principal components explain the majority of the variance in the feature set, we will plot the data in these two dimensions to better assess player similarities.

At first glance, there are differences between players based on overall statistics. For example, LeBron James and James Harden are near each other, indicating star players may be grouped together. There are also similarities based on player position. For example, Centers/Power Forwards such as Anthony Davis and Karl-Anthony Towns are in the bottom of the chart.

```r
# Labels: Players who played more than 36 min per game or less than 3 min per game
labels_pca <- ifelse(nba$MP_pg >= 36 | nba$MP_pg <= 3,
                     as.character(nba$Player), '')
title_pca <- paste0('PCA: NBA - ', ncol(nba_feat) ,' features')

# Plot first two components with positions
plot_pca(nba_pca, data = nba, colour = 'Pos',
         label = labels_pca, title = title_pca, leg_title = 'Position'
          )
```



Clustering

Hierarchical clustering

The first method of clustering we will try is hierarchical clustering. The dendrogram can help provide a visual aid in the number of clusters we can start to use.

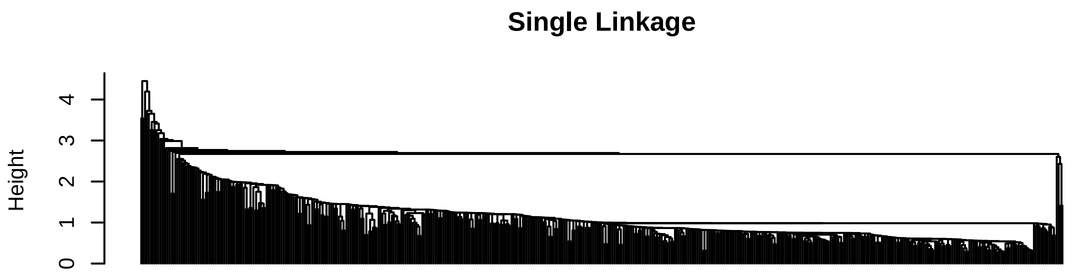
```
# distance matrix for features
nba_dist_sc <- dist(nba_feat_sc, method = 'euclidean')

# try single, centroid, and ward (D2) linkage hier clustering
hcl_single <- hclust(d = nba_dist_sc, method = 'single')
hcl_centroid <- hclust(d = nba_dist_sc, method = 'centroid')
hcl_ward <-  hclust(d = nba_dist_sc, method = 'ward.D2')
```

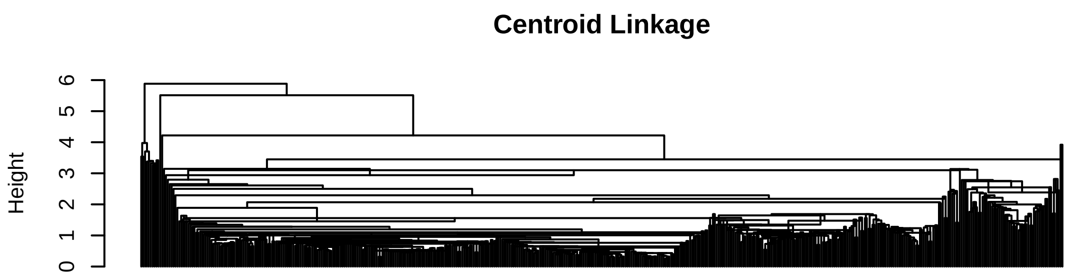
ADJUST WIDTH for PLOTS

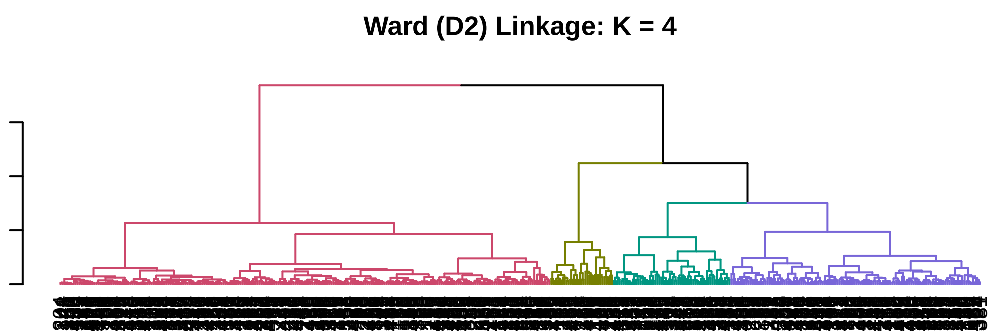
```
par(mfrow = c(3, 1))
# nearest neighbors method
plot(hcl_single, hang = -1, main = 'Single Linkage',
     labels = FALSE, xlab = '', sub = '')
# groups centroid
plot(hcl_centroid, hang = -1, main = 'Centroid Linkage',
     labels = FALSE, xlab = '',  sub = '')
# Ward's minimum variance method,
# with dissimilarities are squared before clustering
```

```r
dend <- as.dendrogram(hcl_ward)
hcl_k <- 4
dend_col <- color_branches(dend, k = hcl_k)
plot(dend_col, main = paste0('Ward (D2) Linkage: K = ', hcl_k),
     labels = FALSE)
```

```
## Warning in plot.window(...): "labels" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "labels" is not a graphical parameter

## Warning in title(...): "labels" is not a graphical parameter
```







Since the Ward dendrogram seems to be the best among the three, we will look at its distribution for 3 and 4 clusters. We chose these initial groupings because this provides a good start to separate and distinguish between player groups.

```
# add cluster labels to main data
nba$hcl_ward_labs_three <- cutree(hcl_ward, k = 3)
nba$hcl_ward_labs_four <- cutree(hcl_ward, k = 4)

hcl_df_three <- data.frame(table(nba$hcl_ward_labs_three))
hcl_df_four <- data.frame(table(nba$hcl_ward_labs_four))
col_names <- c('Clusters', 'Count')
colnames(hcl_df_three) <- col_names
colnames(hcl_df_four) <- col_names


print(kable(hcl_df_three, caption = 'Three Clusters'))
```

```
##
##
## Table: Three Clusters
##
## Clusters    Count
## ---------  ------
## 1             259
## 2             194
## 3              33
```

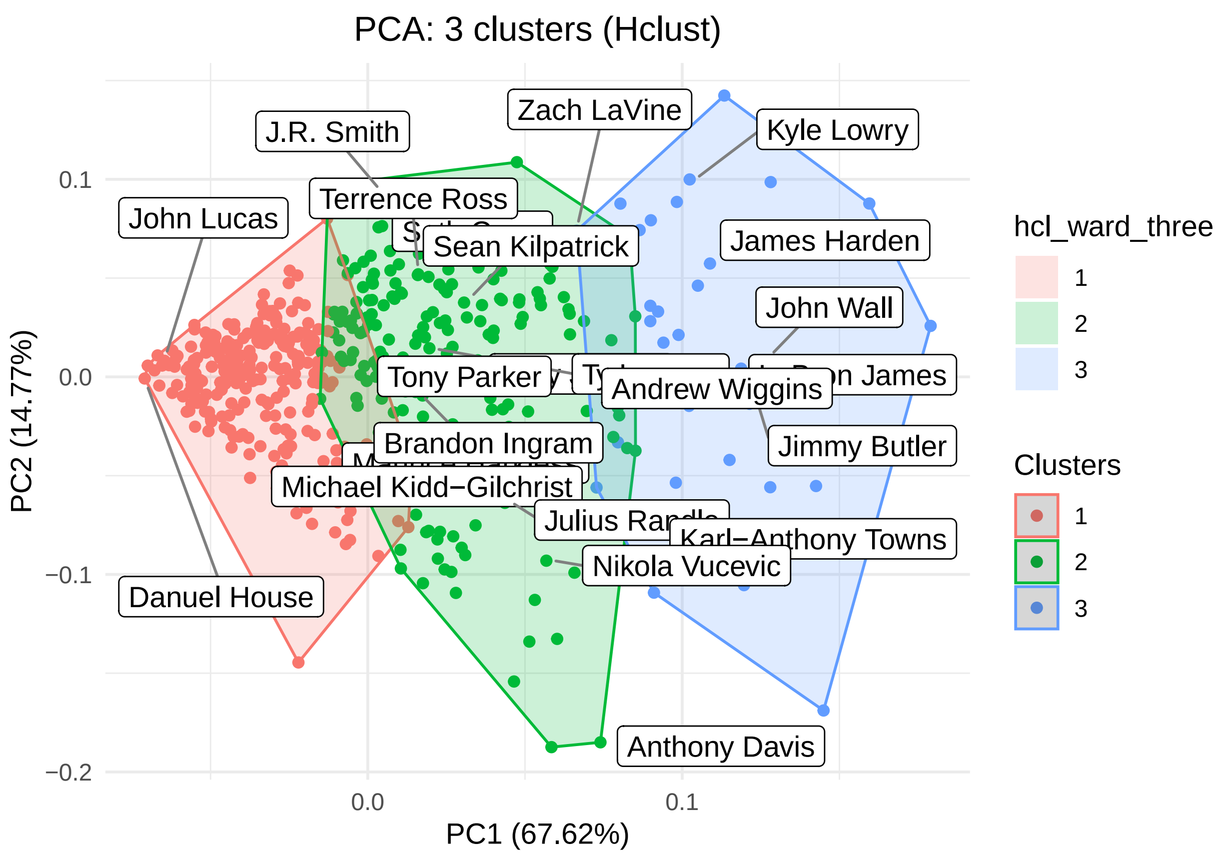
```
print(kable(hcl_df_four, caption = 'Four Clusters'))
```

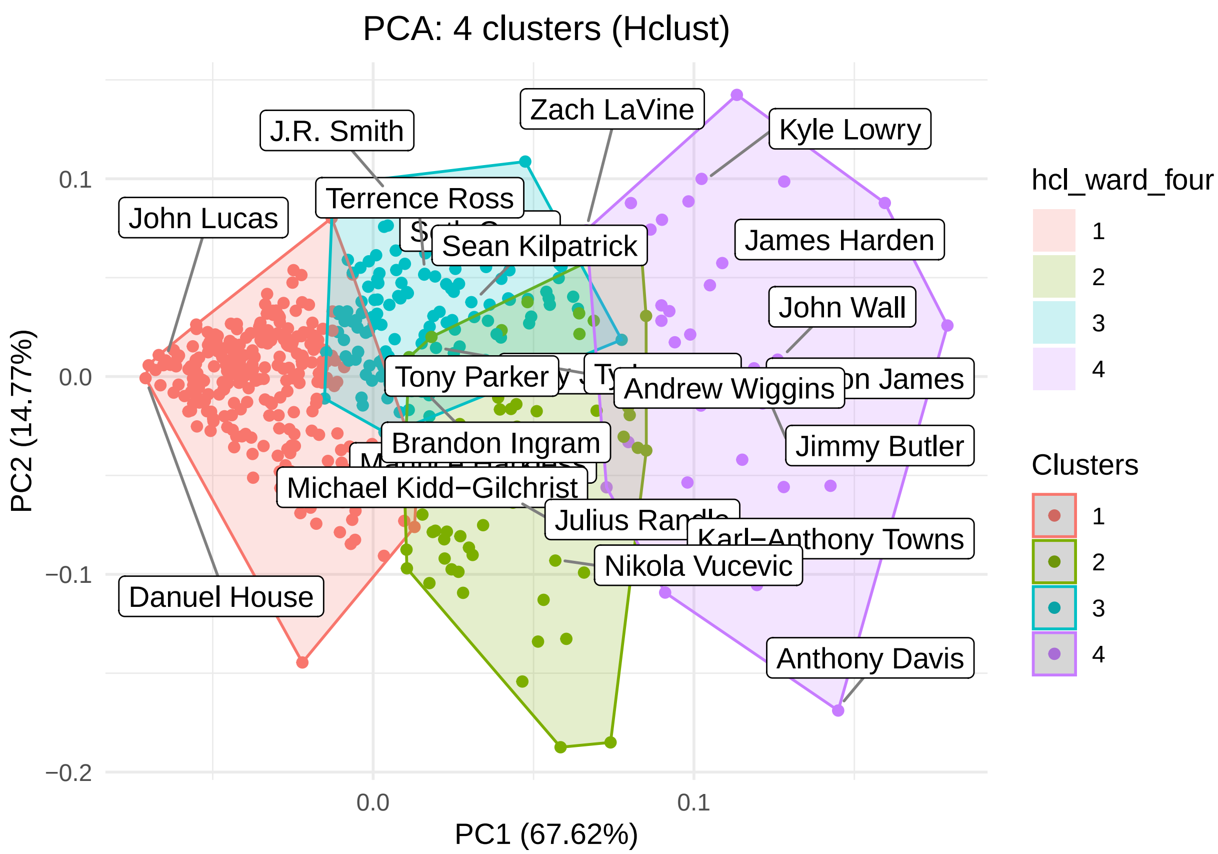
```
##
##
## Table: Four Clusters
##
## Clusters    Count
## ---------  ------
## 1             259
## 2              62
## 3             132
## 4              33
```

Visualizing the clusters in PC space, we see clear separation in the 4-cluster solution vs the 3-cluster method based on apparent skillsets. For example, LeBron James and James Harden (star players) are in one cluster, whereas John Lucas and Danuel House (lower performers) are in the left-most cluster, opposite to the 'star player' cluster on the right. These will be explored further when we look at player statistics by cluster.

```
# add labels to data
nba$hcl_ward_three <- factor(cutree(hcl_ward, k = 3))
nba$hcl_ward_four <- factor(cutree(hcl_ward, k = 4))

# player names to include in plot
hcl_labels <- ifelse(nba$MP_pg >= 36 | nba$MP_pg <= 2.5 |
                     (nba$MP_pg >= 28.8 & nba$MP_pg <= 29) |
                     (nba$MP_pg >= 25 & nba$MP_pg <= 25.2),
                     as.character(nba$Player), '' )

# elements to loop over
```

```r
hcl_labs <- names(nba %>% select(tail(names(.), 2)))
hcl_ks <- c(3, 4)

# plot hclust labels superimposed over PCA
for (i in seq_along(hcl_labs)) {
  p <- plot_pca(nba_pca, frame = TRUE,
                data = nba, colour = hcl_labs[i],
                title = paste0('PCA: ', hcl_ks[i], ' clusters (Hclust)'),
                label = hcl_labels,
                leg_title = 'Clusters'
  )

  print(p)
}
```





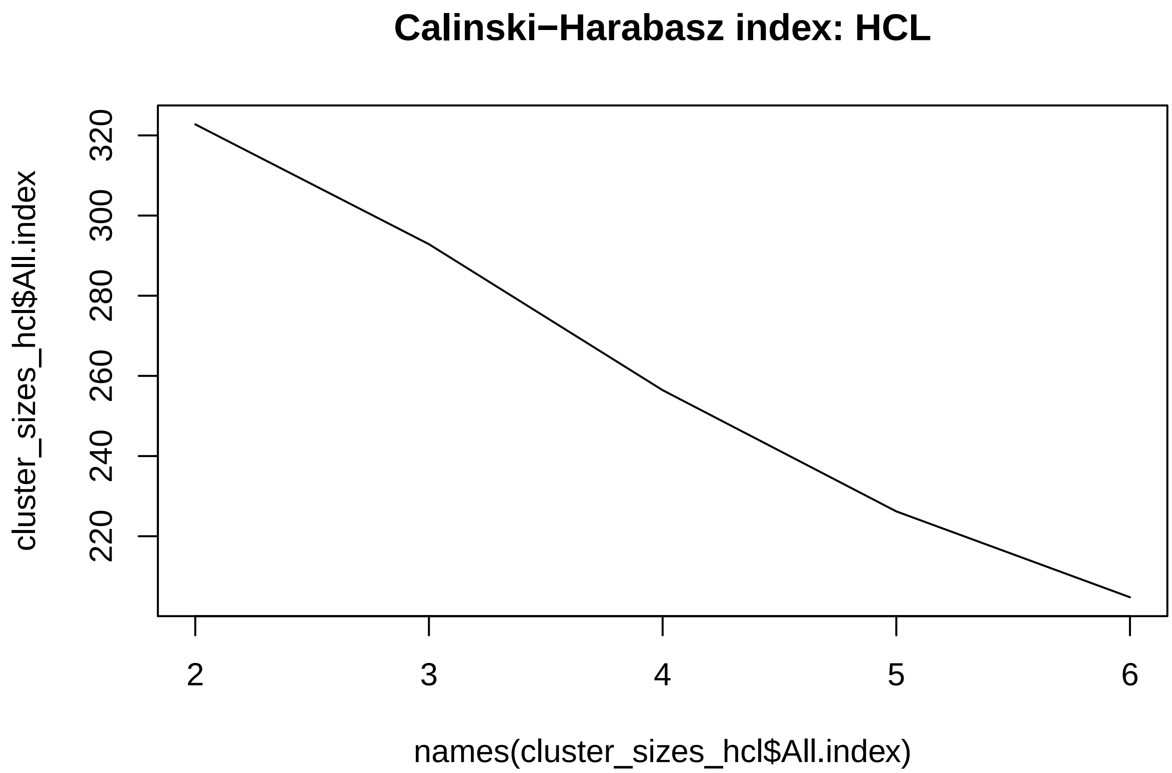
Optimize number of clusters

To optimize the number of clusters, we used two index methods: Calinski-Harabasz and Silhouette. These methods measure how well separated clusters are, and how homogenous data is within each cluster. Both indices yield 2 as the optimal number.

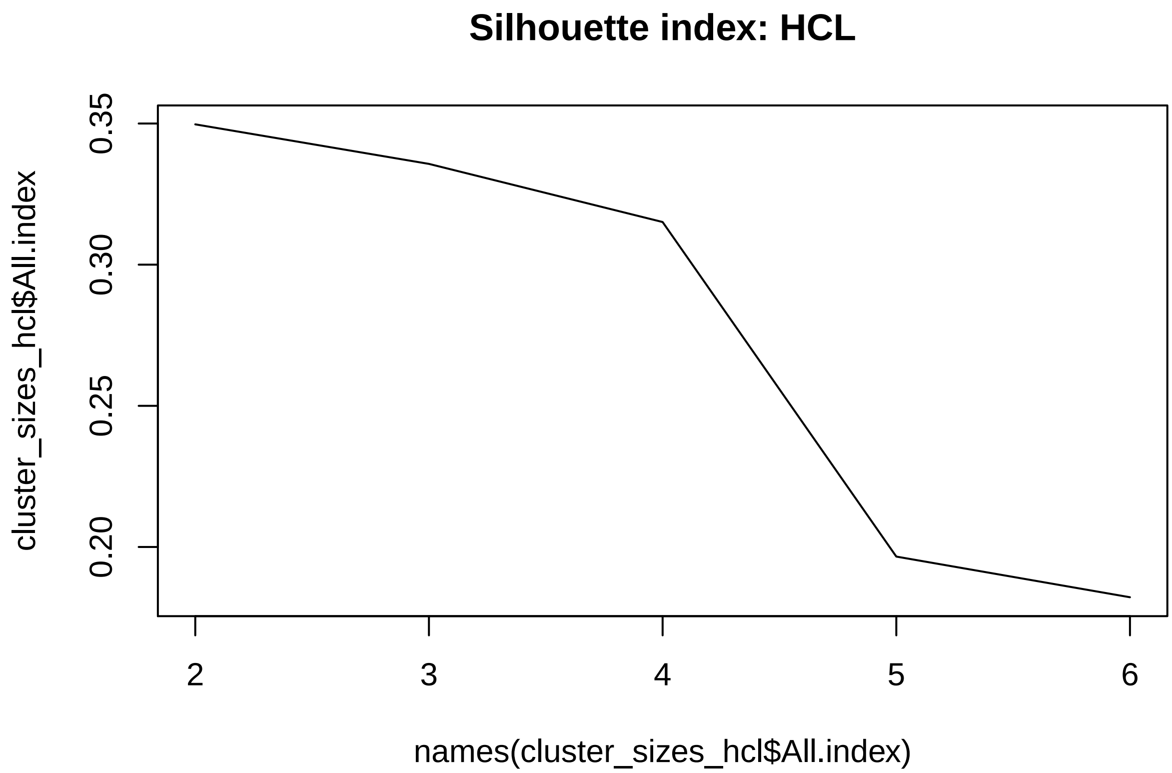
Although the indices say 2 is optimal, these does not provide enough distinction or separation among players. However, the Silhouette index provides 4 as a local maximum. This provides a more practical distinction among players.

Methods: Calinski-Harabasz index and Silhouette

```r
# get optimal cluster sizes
cluster_sizes_hcl <- NbClust(data = nba_feat_sc,
                             # it will likely be harder to interpret clusters
                             # past this amount
                             max.nc = 6,
                             method = 'ward.D2',
                             index = 'ch')

# plot C(G)
plot(names(cluster_sizes_hcl$All.index),
     cluster_sizes_hcl$All.index,
     main = 'Calinski-Harabasz index: HCL',
     type = 'l')
```




```r
# get optimal cluster sizes
cluster_sizes_hcl <- NbClust(data = nba_feat_sc,
                             # it will likely be harder to interpret clusters
                             # past this amount
                             max.nc = 6,
                             method = 'ward.D2',
                             index = 'silhouette')

# plot C(G)
plot(names(cluster_sizes_hcl$All.index),
     cluster_sizes_hcl$All.index,
     main = 'Silhouette index: HCL',
     type = 'l')
```



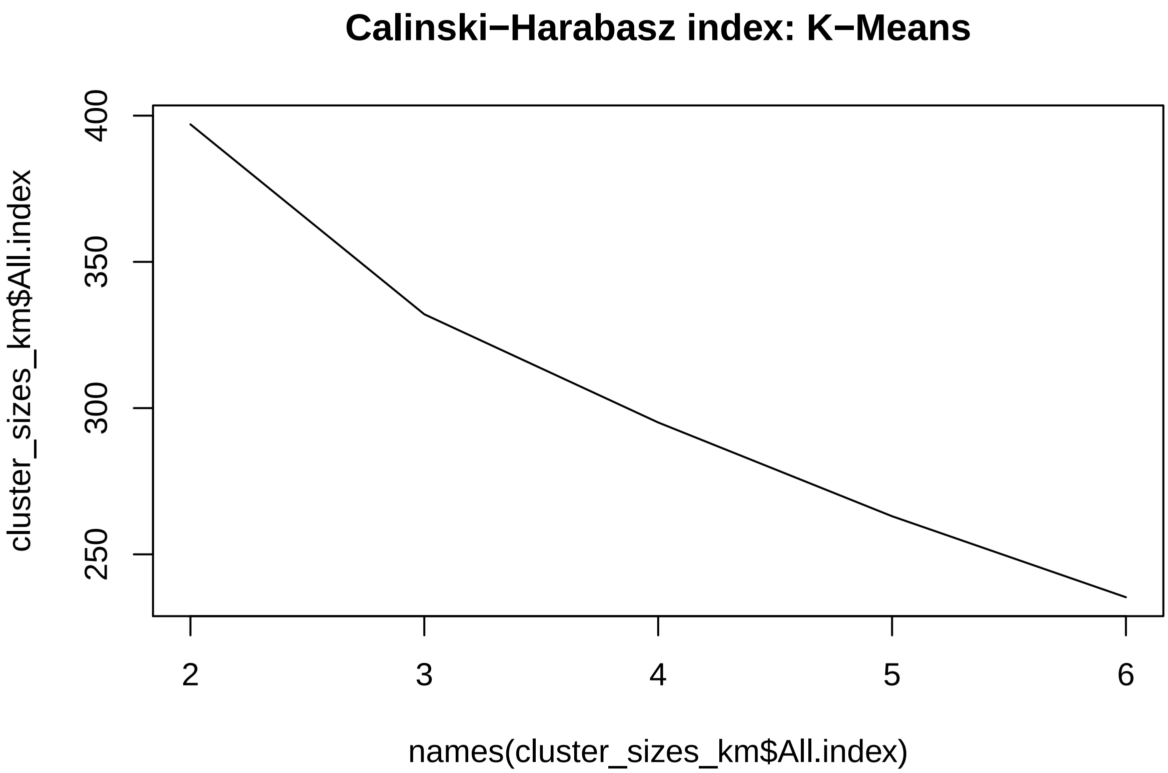
K-Means

Optimize number of clusters

Similar to the optimization for hierarchical, we found similar results for K-Means. We decided with the 4-cluster solution based on the Silhouette index (local maximum.)

Method: Calinski-Harabasz index

```
# get optimal cluster sizes
cluster_sizes_km <- NbClust(data = nba_feat_sc,
                            # it will likely be harder to interpret clusters
                            # past this amount
                            max.nc = 6,
                            method = 'kmeans',
                            index = 'ch')

# plot C(G)
plot(names(cluster_sizes_km$All.index),
     cluster_sizes_km$All.index,
     main = 'Calinski-Harabasz index: K-Means',
     type = 'l')
```



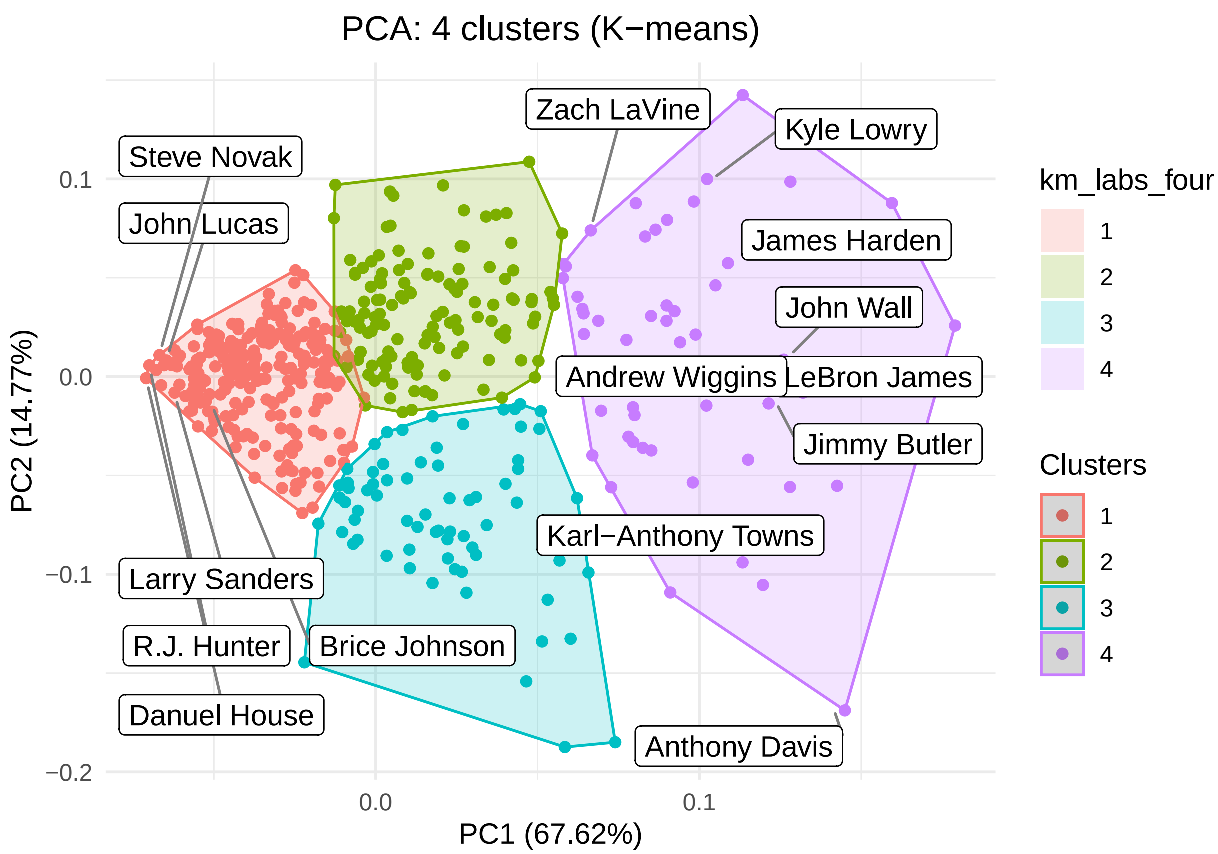

```r
# get optimal cluster sizes
cluster_sizes_km <- NbClust(data = nba_feat_sc,
                            # it will likely be harder to interpret clusters
                            # past this amount
                            max.nc = 6,
                            method = 'kmeans',
                            index = 'silhouette')

# plot C(G)
plot(names(cluster_sizes_km$All.index),
     cluster_sizes_km$All.index,
     main = 'Silhouette index: K-Means',
     type = 'l')
```

K-means clustering with 4 groups

```
km_k <- 4
km_four <- kmeans(x = nba_feat_sc,
            centers = km_k,
            nstart = 100,
            algorithm = 'Hartigan-Wong')

nba$km_labs_four <- factor(km_four$cluster)
```

Compared to the hierarchical solution presented earlier, there is cleaner separation in the K-Means plot. We will see how these clusters are separated by inspecting features within each group.

```
# plot k-means clusters in PC space
# Labels: Players who played more than 36 min per game or less than 3 min per game
km_labels <- ifelse(nba$MP_pg >= 36 | nba$MP_pg <= 3,
                         as.character(nba$Player), '' )

plot_pca(km_five, data = nba, frame = TRUE, colour = 'km_labs_four',
          title = paste0('PCA: ', km_k, ' clusters (K-means)'),
          label = km_labels, leg_title = 'Clusters')
```



One explanation for the imbalanced number of players across clusters is that player skillset level is inherently imbalanced. This imbalance is reflected in the univariate plots, where most densities were positively skewed. This suggests that there a few players whose statistics significantly exceed those of the average player. This is also reflected in the average statistics by cluster shown below.

Table 2: K-Means Player Distribution

Clusters	Count
1	236
2	130
3	69
4	51

```
# averages by cluster
nba_km_avg <- data.frame(nba
                         %>% select(km_labs_four, MP_pg, PTS_pg, TRB_pg,
                                    AST_pg, BLK_pg, STL_pg, VORP, PER, RPM)
                         %>% group_by(km_labs_four)
                         %>% summarise_all(list(mean))
                         )

kable(nba_km_avg, caption = 'Average Stats by Cluster')
```

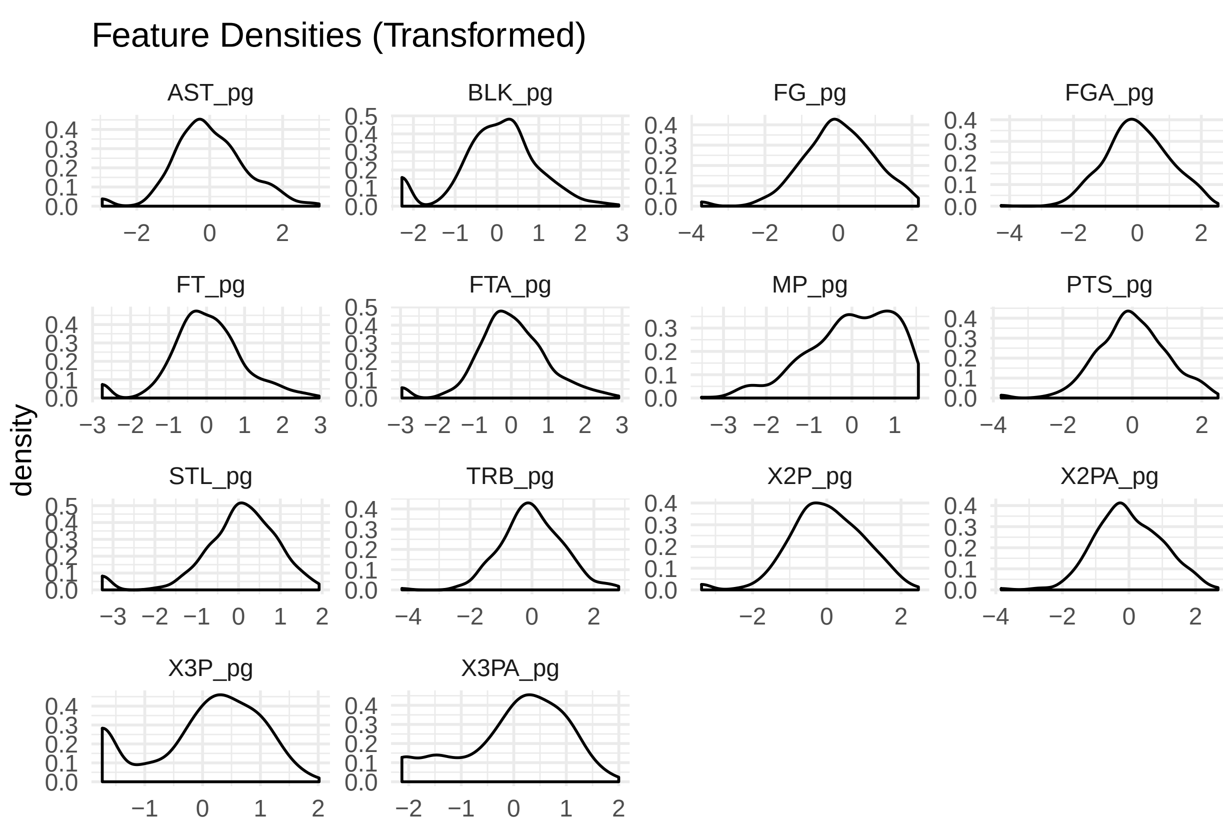
Table 3: Average Stats by Cluster

km_labs_four	MP_pg	PTS_pg	TRB_pg	AST_pg	BLK_pg	STL_pg	VORP	PER	RPM
1	12.24558	3.94709	2.129925	0.8788919	0.2354097	0.3548853	-0.0669492	10.18941	-1.9458898
2	25.61995	10.30825	3.484223	2.5209841	0.2885179	0.8214061	0.5015385	12.75308	-0.8625385
3	24.96591	10.29999	7.010810	1.6435808	0.9473312	0.7868895	1.2681159	17.09710	0.3679710
4	33.85906	21.82156	5.724658	4.7255814	0.6158290	1.1623242	3.1921569	21.29020	2.5100000

Model-Based Clustering

To test alternative approaches to Hierarchical and K-Means clustering, we will perform model-based clustering. Prior to modeling, we will transform the data using the cube root. This normalizes the data, which is a key assumption in model-based clustering. We tested other transformations such as the square root, Box-Cox, and log + 1. Of these, The cube root yielded the best transformation for model-based clustering.

```
# transform features
nba_feat_cr <- (nba_feat) ^ (1/3)
# scale transformed features
nba_feat_cr_sc <- scale(nba_feat_cr)
```

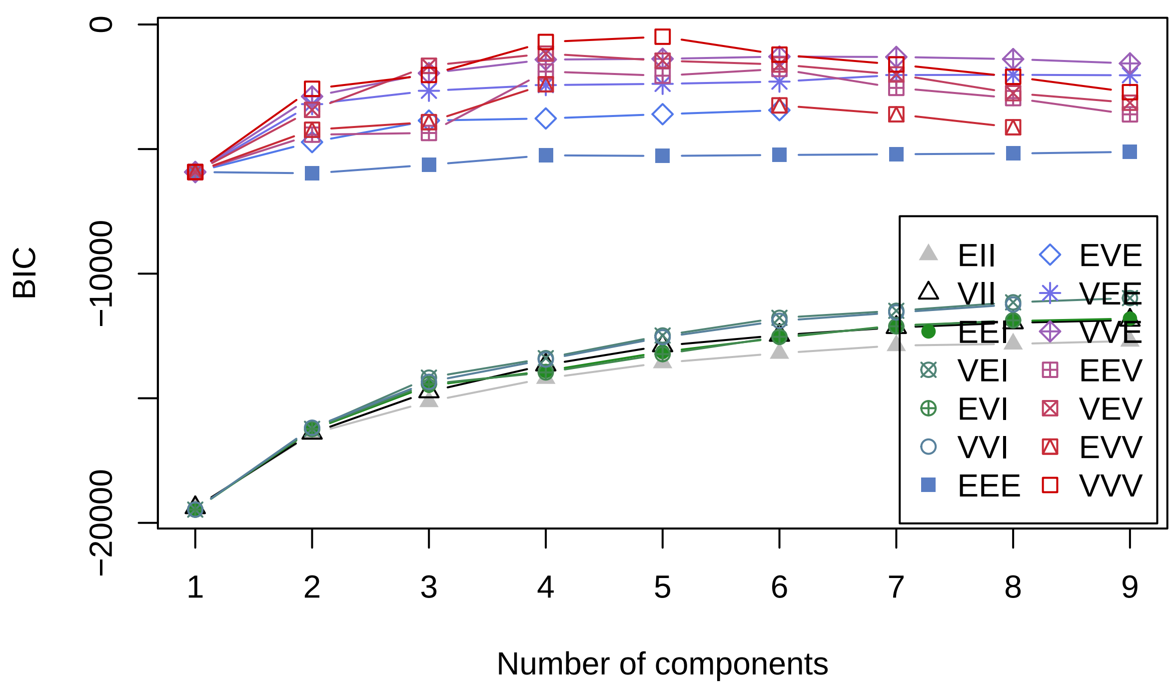


The model-based clustering produced a five-cluster solution based on the BIC. Compared to the previous methods, model-based clustering tended to group players by moreseo by style of play. On the other hand, K-Means and Hierarchical appear to cluster on overall skillsets across statistics. The clusters also align more with the original PCA plot by position. This type of clustering suggests emphasis on style of play, e.g. better rebounders/blockers, better passers, scorers, etc.

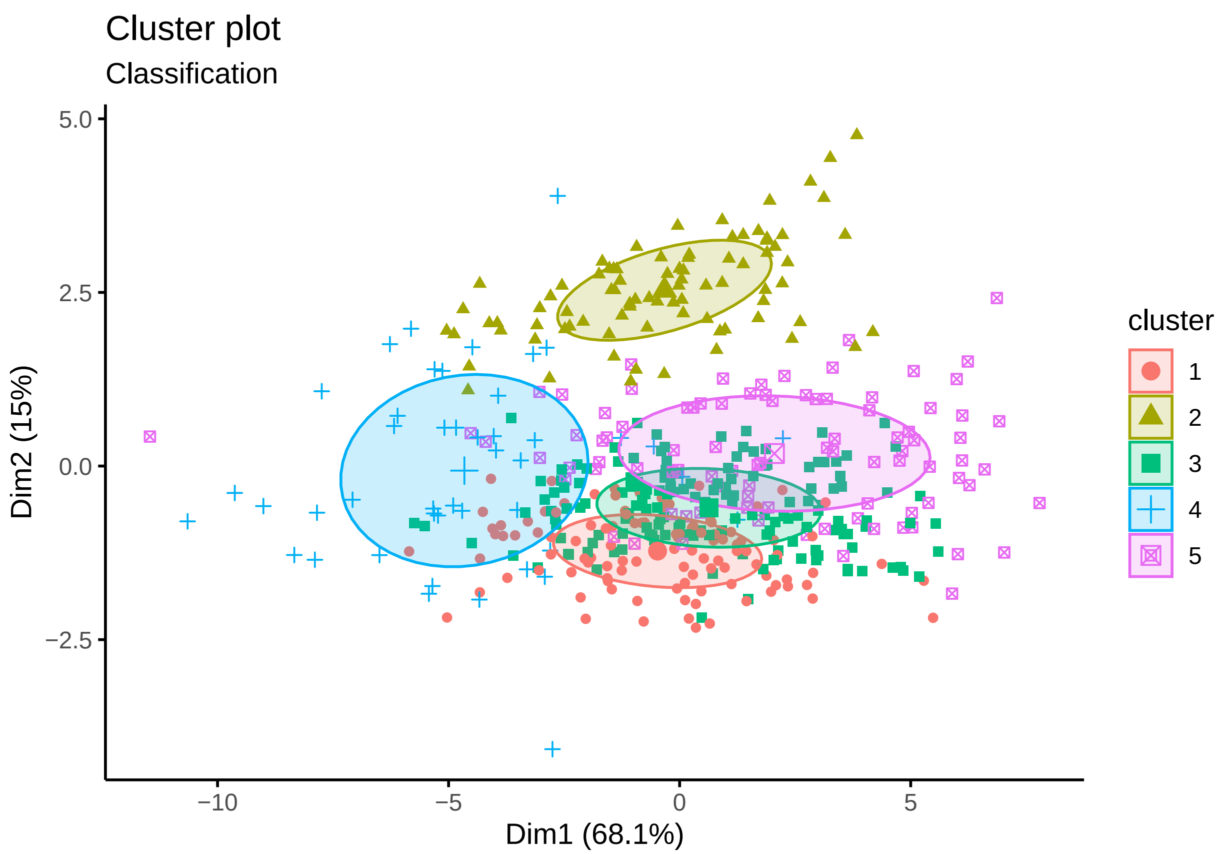
```r
# run model
player_clust.mcl <- Mclust(nba_feat_cr_sc)
summary(player_clust.mcl)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model
## with 5 components:
##
##  log-likelihood   n  df       BIC       ICL
##        1607.448 486 599 -490.6436 -505.4606
##
## Clustering table:
##   1   2   3   4   5
## 106  84 160  45  91
```

```r
plot(player_clust.mcl, what = "BIC")
```




```r
# plot results
fviz_mclust(player_clust.mcl, "classification", geom = "point")
```




```r
# add cluster labels to plot
nba$mcl_labs <- player_clust.mcl$classification
```

Compare methods between Clusters

Between HCL and KM, the maximum possible agreement between clusters is 87% (424 / 486). Between HCL and MCL, the maximum possible agreement between clusters is 30% (148 / 486). Between KM and MCL, the maximum possible agreement between clusters is 41% (201 / 486).

Based on the analysis, MCL tends to group players by position, whereas HCL and KM tend to cluster based on overall player statistics. This conclusion was reached based on inspecting distributions of player positions across the clustering methods. Because we are looking at player statistics and team compensation, the model-based clustering is not an ideal fit for this purpose.

```r
# run crosstabs between cluster methods
xtab_hcl_km <- xtabs(~nba$hcl_ward_four + nba$km_labs_four)
xtab_hcl_mcl <- xtabs(~nba$hcl_ward_labs_four + nba$mcl_labs)
xtab_km_mcl <- xtabs(~nba$km_labs_four + nba$mcl_labs)

xtab_hcl_km
```

```
##                   nba$km_labs_four
## nba$hcl_ward_four   1   2   3   4
##                 1 231   4  24   0
##                 2   0   8  42  12
##                 3   5 118   3   6
##                 4   0   0   0  33
```

```
xtab_hcl_mcl
```

```
##                         nba$mcl_labs
## nba$hcl_ward_labs_four  1  2  3  4  5
##                       1 55 60 75 41 28
##                       2  0 24 12  1 25
##                       3 48  0 63  3 18
##                       4  3  0 10  0 20
```

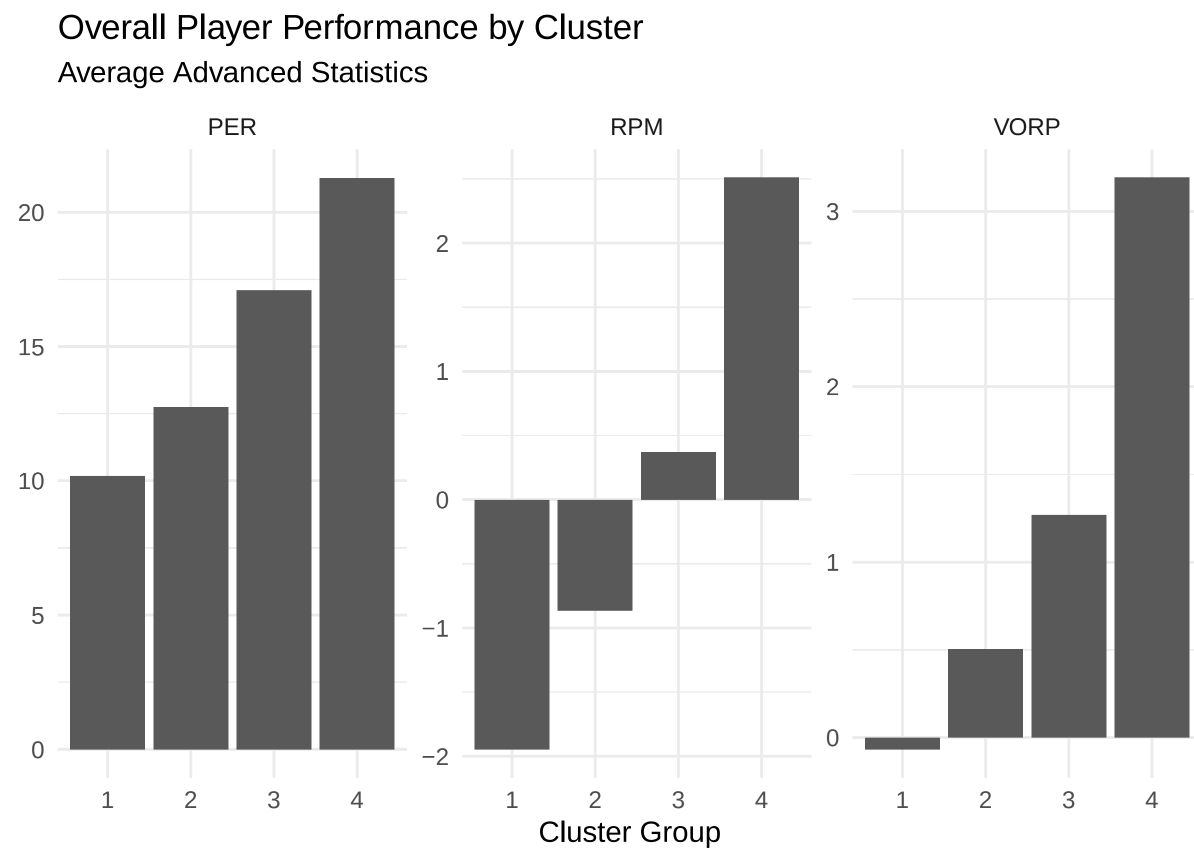
```
xtab_km_mcl
```

```
##                   nba$mcl_labs
## nba$km_labs_four  1  2  3  4  5
##                 1 52 40 74 41 29
##                 2 51  0 63  2 14
##                 3  0 44  7  2 16
##                 4  3  0 16  0 32
```

Final Cluster selection

K-Means (4 clusters) was the optimal solution. Comparing the HCL and KM cluster plots (per above) reveals the K-Means produces clearer separation of players based on overall skillsets.

We validated this by comparing the clusters against advanced statistics. PER, VORP, and RPM are advanced statistics commonly used to assess general player performance. None of these statistics were used in the cluster



modeling.

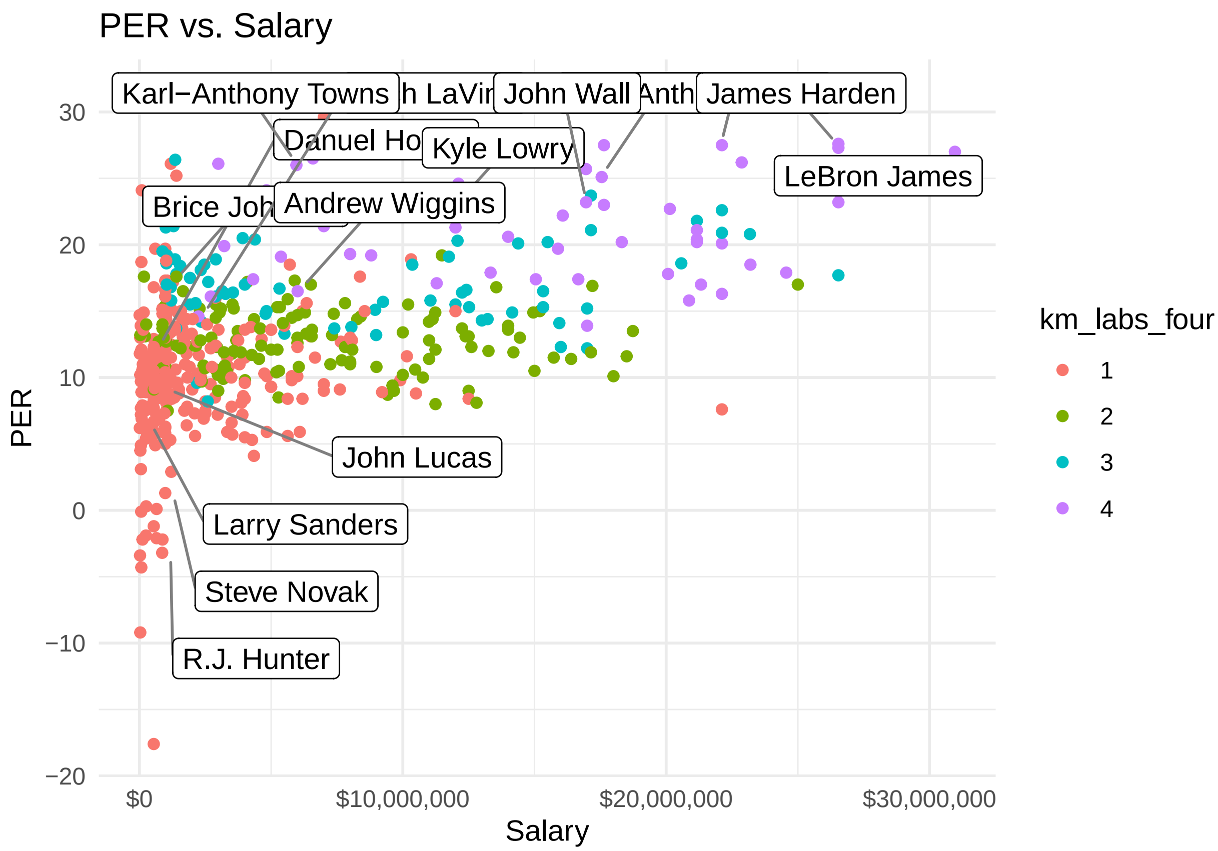
Post-Cluster Analysis

We will now look at different statistics and demographics to assess cluster membership and draw insights.

```
### Star, Role, Bench, Low Performer
```

There is potential to update salaries based on player tiers. For example, Chandler Parsons was paid 22M but is considered a low performer, and is paid more than star players such as Steph Curry (12M) and Kawhi Leonard (17.6M). This shows that salary is not necessarily strongly correlated with player performance.

Clusters vs. Player Salaries




```r
# Highest Paid players in Lowest Tier
kable(head(data.frame(nba
          %>% select(Player, G, MP_pg, Tm,
                     Salary, PER, cl_label)
          %>% filter(km_labs_four == 1)
          %>% arrange(desc(Salary))
          )), caption = 'Highest Paid Players: Lower Performances')
```

Table 4: Highest Paid Players: Lower Performances

Player	G	MP_pg	Tm	Salary	PER	km_labs_four
Chandler Parsons	34	19.85294	MEM	22116750	7.6	1
Miles Plumlee	45	10.75556	MIL	12500000	8.4	1
Amir Johnson	80	20.10000	BOS	12000000	15.0	1
Mirza Teletovic	70	16.18571	MIL	10500000	8.8	1

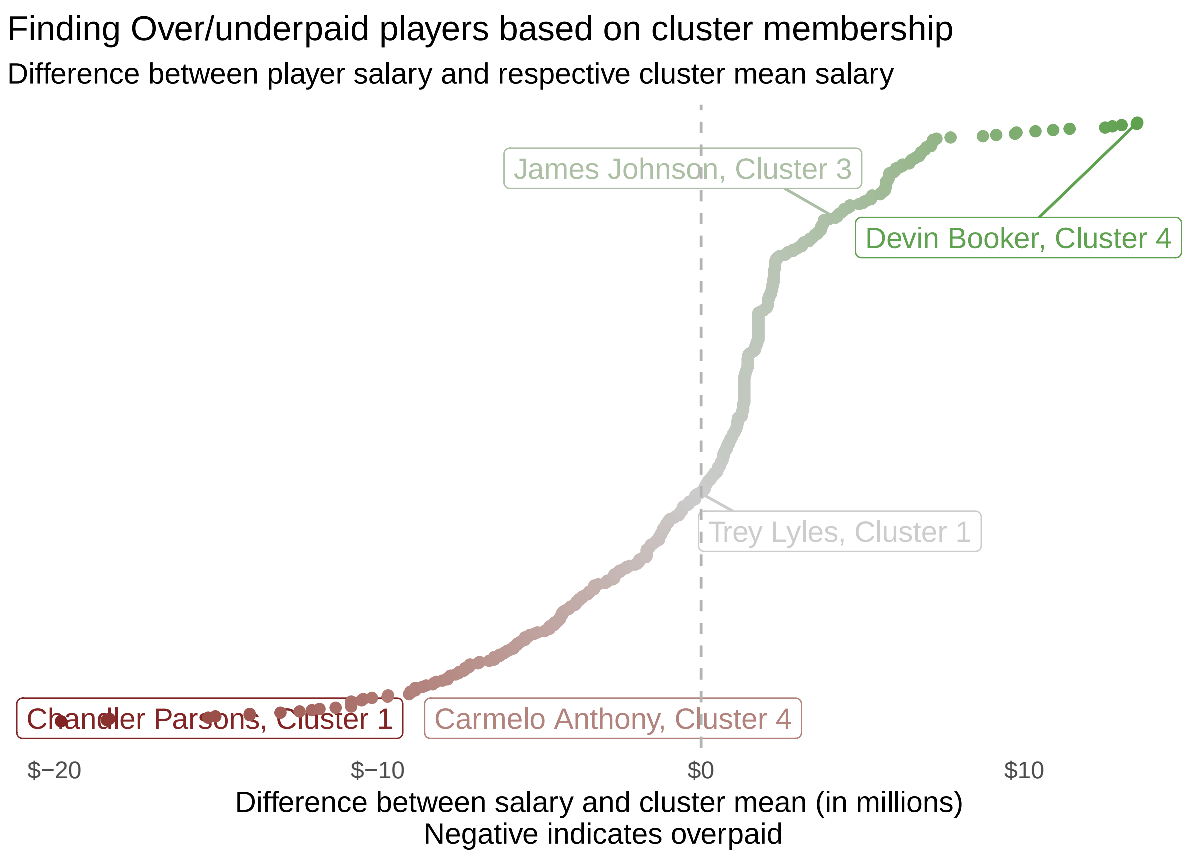
Player	G	MP_pg	Tm	Salary	PER	km_labs_four
Al Jefferson	66	14.10606	IND	10314532	18.9	1
Alec Burks	42	15.54762	UTA	10154495	11.6	1

```
# Lowest Paid players in highest tier
kable(head(data.frame(nba
          %>% select(Player, G, MP_pg, Tm, Salary, PER, cl_label)
          %>% filter(km_labs_four == 4)
          %>% arrange(Salary)
          )), caption = 'Lowest Paid Players: Star Players')
```

Table 5: Lowest Paid Players: Star Players

Player	G	MP_pg	Tm	Salary	PER	km_labs_four
Devin Booker	78	35.00000	PHO	2223600	14.6	4
Zach LaVine	47	37.21277	MIN	2240880	14.6	4
Dennis Schroder	79	31.45570	ATL	2708582	16.1	4
Giannis Antetokounmpo	80	35.56250	MIL	2995421	26.1	4
C.J. McCollum	80	34.95000	POR	3219579	19.9	4
Kristaps Porzingis	66	32.78788	NYK	4317720	17.4	4

The chart below indicates how much a player is overpaid or underpaid with respect to their cluster average salary. For example, Chandler Parsons was paid 22M vs. the cluster salary average (~2M), indicating he was overpaid. It is important to note that there are players who may have been injured, so their statistics may not be commensurate to their salaries.



Team Compensation and Performance vs clusters

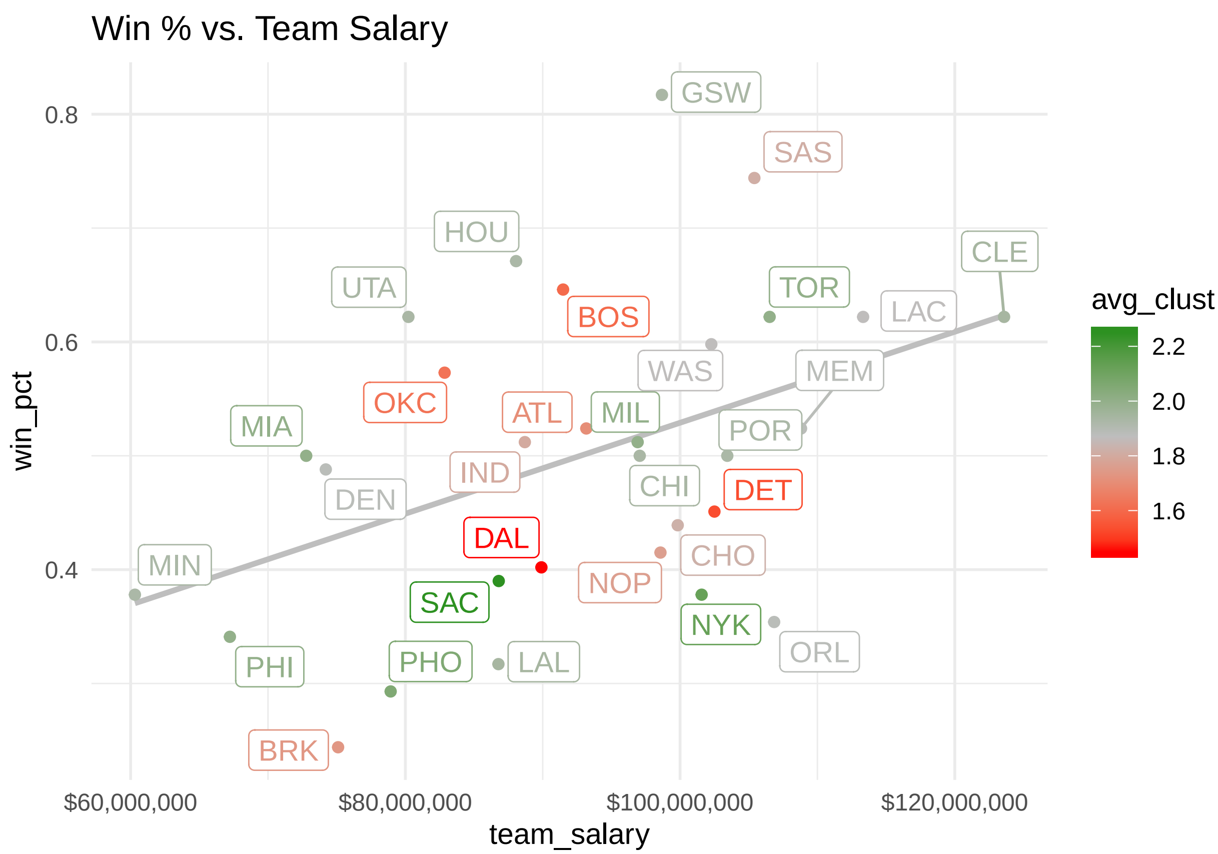
```
# convert labels to numeric
nba$cl_lab_numeric <- as.numeric(nba[ , cl_label])
nba_team <- data.frame(nba
                       %>% select(Tm, Salary, win_pct, cl_lab_numeric)
                       %>% group_by(Tm)
                       %>% summarise(team_salary = sum(Salary),
                                     win_pct = mean(win_pct),
                                     avg_clust = mean(cl_lab_numeric))
                       )

# order by descnding average cluster label
arrange(nba_team, desc(avg_clust))
```

```
##     Tm team_salary win_pct avg_clust
## 1  SAC    86799609   0.390  2.250000
## 2  NYK   101570502   0.378  2.125000
## 3  PHO    78930157   0.293  2.055556
## 4  MIA    72782449   0.500  2.000000
## 5  MIL    96913241   0.512  2.000000
## 6  PHI    67225712   0.341  2.000000
## 7  TOR   106521470   0.622  2.000000
## 8  CLE   123591014   0.622  1.941176
## 9  LAL    86775415   0.317  1.941176
## 10 CHI    97064073   0.500  1.933333
```

```
## 11 GSW    98681493   0.817  1.933333
## 12 UTA    80223193   0.622  1.933333
## 13 HOU    88062247   0.671  1.928571
## 14 MIN    60311572   0.378  1.928571
## 15 POR   103439444   0.500  1.928571
## 16 DEN    74208517   0.488  1.882353
## 17 MEM   108808118   0.524  1.882353
## 18 ORL   106849160   0.354  1.882353
## 19 LAC   113327068   0.622  1.866667
## 20 WAS   102276673   0.598  1.866667
## 21 CHO    99830531   0.439  1.823529
## 22 SAS   105410231   0.744  1.812500
## 23 IND    88698690   0.512  1.800000
## 24 NOP    98573436   0.415  1.761905
## 25 BRK    75102568   0.244  1.736842
## 26 ATL    93172774   0.524  1.705882
## 27 OKC    82858524   0.573  1.625000
## 28 BOS    91484921   0.646  1.600000
## 29 DET   102503259   0.451  1.533333
## 30 DAL    89904500   0.402  1.450000
```

Although a team can have better players on average clusters, there are many variables at play here. A team can be better on average but poor management or coaching can affect a team's overall performance, e.g. NYK. Interestingly, GSW did not have the highest average cluster rating, because their bench is not very strong. This speaks to the strong influence that starter players can have on team performance. Another interesting note is that teams can play well even if they do not have many all-stars or a strong overall team, e.g BOS. This could be driven by great coaching and team chemistry. It is important to note that items such as injuries could greatly influence win %, even if players have high ratings.

Although there is a correlation between overall team salary and win %, it is interesting that average player rating does not necessarily align with overall win %.




```
# Inspect some teams
teams_sample <- c('NYK', 'GSW', 'BOS')

teams_sample_list <- list(rep(NA, length = length(teams_sample)))
for (i in seq_along(teams_sample)) {
  teams_sample_list[[i]] <- nba[nba$Tm == teams_sample[i],
                                c('Player', 'PER', cl_label)]
}

# change index to see different teams
teams_sample_list[[2]]
```

```
##               Player  PER km_labs_four
## 82         Ian Clark 13.1            1
## 98     Stephen Curry 24.6            4
## 119     Kevin Durant 27.6            4
## 164   Draymond Green 16.5            3
## 211   Andre Iguodala 14.4            2
## 235     Damian Jones  5.3            1
## 268 Shaun Livingston 10.1            1
## 270     Kevon Looney 13.4            1
## 285    James Michael 13.0            1
## 286    Patrick McCaw  8.6            1
## 293     JaVale McGee 25.2            1
## 344    Zaza Pachulia 16.1            3
## 427    Klay Thompson 17.4            4
```

```
## 443 Anderson Varejao  9.4             1
## 457       David West 16.6             1
```