NBA Data Analysis

Chenjie Li, Dilruba Palabiyik, Qiao Qiao

***Abstract*— In this report, using the NBA data collected from past 10 seasons (from season $2009-2010$ to season $2018-2019$), we try to find the most important statistics playing big difference in teams' success. Also, by redefining the traditional 5 types of positions in the lineup, we find the patterns of the distributions of the redefined roles and some interesting developments of some players over the years.**

I. Introduction

As a famous professional basketball league, the National Basketball Association (NBA) has drawn the enthusiasm and attention from millions of fans all over the world. Besides being entertaining to fans and audiences, the performance of the NBA teams is of significant financial interests to the team owners and management. As the technology advances, more and more advanced statistics have emerged into our sight. For the players, Hollinger's PER [1], which is a quite objective and comprehensive rating metric for evaluating a player's overall value on the court. For teams, we also have very informative formula such as offensive rating, defensive rating [2] ... In this report, we focus on two specific topics. First, what are the most important statistics contributing to teams' success and failure, in which we use different linear models and choose the one with the best accuracy. Second, we try to use clustering approach to redefine the roles of modern NBA, the break down the trend of the game today compared with the trend 10 years ago.

II. Data Source

There are many data resources available for us to choose from, like the NBA official website [2], basketball reference [3]. Our data source is mainly from play by play website [4], which is by far one of the most comprehensive and structured data source we found. It not only contains the basic stats charts, but also brilliant visualizations. Also, since [4] doesn't have team rankings, we got the ranking information from ESPN NBA website [5].

To be able to work with the data from the website, we developed some *Web Scrapers* in Python. For Team rankings, we chose to use *Scrapy* library because of its speed and performance. For [4], we chose to use *Selenium* since this website is mainly developed using AJAX requests, and *Selenium* can easily handle those elements by locating tags using XPATH and virtually perform the "select" and "click" action.

For our data storage, we chose to put scraped data in a Database for our convenience. In this project, we use Postgres Database.

The data we collected can be divided into two categories: Player and Team. And within each category, we have scoring, assists and rebounds.

III. Data Cleaning

A. Collinearity

Collinearity, refers to the situation in which two or more predictor variables are closely related to each other. [6]. In the regression context, if two predictors are highly correlated, it will be very difficult to seperate out the individual effects of collinear variables on the response. In our project, since we are handling high dimensional data (over 50 predictors), the collinearity problem is extreme: any variable in the model can be written as a linear combination of all of the other variables in the model. So it makes sense for us to put *minimize multicollinearity* as our first step.

Fig. 1 shows the correlation matrix in R of part of our predicors: There are a lot of highly(> 0.8) correlated variables because some statistics are either hierarchical related or used to describe the same situation. e.g. *rebounds* and *defensive rebounds* are hierachical correlated. Also, *FG2M(2 Pont Field Goals Made)* is highly correlated with *FG2A(2 Point Field Goals Attempt)* because that it is usually the case that the more attempts you make, the higher number of mades you will get. Based on the domain knowledge and by observing the result of the *Correlation Matrix*, we did some removal of the redundant predictors. The result is shown in Fig. 2.

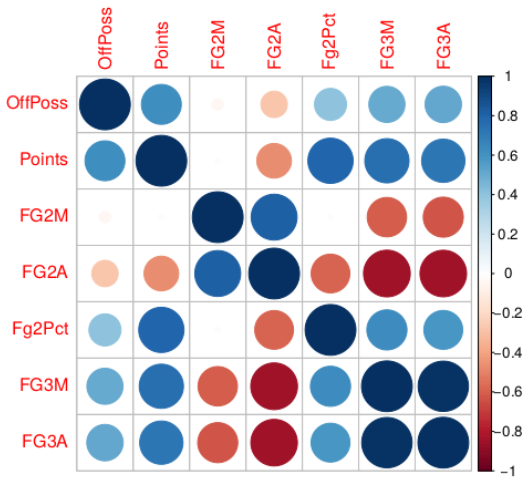

Fig. 1. Correlation Matrix Before Removal Of Redundant Predictors

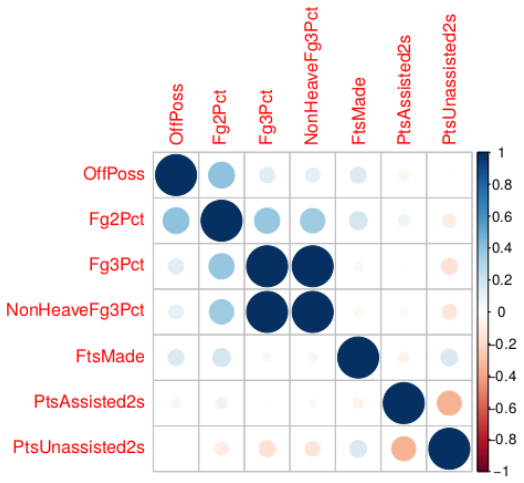

Fig. 2. Correlation Matrix After Removal Of Redundant Predictors

B. Outliers

In statistics, an outlier is a data point that differs significantly from other observations[7]. An outlier may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. An outlier can cause serious problems in statistical analyses.

In our setting, in team level we didn't remove any outliers because the "treatment" of each team in every season is the same. But in Section V, we considered the outliers' effects. This is because the players are less restricted in a lot of ways. For example, a player could only play 10 games in whole season (each team plays 82 games per season). Also, a player could player a decent number of games but was only on the court for 5 miniutes per game (1 game is 48 minutes). Based on the number of games played(GP) and number of minutes per game (MP), we "filtered out" some players whose statistics may not convey his true value based on the data we collected.

The rest of the report is structured as follows: Section IV describe several linear models we tried to find the most important indexes in teams' success. Section V shows the process of player clustering and the interesting findings based on that. Section VI summarize the work we did and list the conclusions we drew based our analysis and visualizations.

IV. Supervised Learning on Teams

In this section, we will use multiple supervised learning methods to analysis teams' dataset.

The statistical model we have used works on analyzes the seasonal level data. In other words, each data entry in our study is a team's total number of games won in that regular season, and the accumulative statistics from the game logs of the season and some descriptive features of the team. Hence we filter out some random events that happened to a team because players and coaches switched teams.

In a regular season, a team plays 82 games against different opposing teams following the NBA schedule. We denote GW as the number of games won by the team out of the 82 games. Needless to say, GW is the most important number that directly represents the performance of a team in a season. But it is the consequence of many factors. Naturally, we put GW as the dependent variable in the predictive model.

A. K-Fold Cross Validation

For all the models we use in this part, we compare the module performances by using Cross Validation, which, in essense, is just a systematic evaluation by implementing "train and test" procedure. We splitted our data into 'k' mutually exclusive random sample portions. Each time, we keep one of the portions as test data, we build the model on the remaining (k-1 portion) data and calculate the mean squared error of the predictions. This is done for each k random samples. We chose createDataPartition in R to perform an 80/20 test-train split (80% training and 20% testing).

B. Linear regression

We first use linear regression to see the performance of the model.

```
## Residual standard error: 0.08425 on 193 degrees of freedom
## Multiple R-squared:  0.7495, Adjusted R-squared:  0.7014
## F-statistic:  15.6 on 37 and 193 DF,  p-value: < 2.2e-16
```

Fig. 3. Result of Linear Regression

```
Confusion Matrix and Statistics

          true
prediction  0  1
         0 50  2
         1  2  2

               Accuracy : 0.9286
                 95% CI : (0.8271, 0.9802)
    No Information Rate : 0.9286
    P-Value [Acc > NIR] : 0.6289

                  Kappa : 0.4615
 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.9615
            Specificity : 0.5000
         Pos Pred Value : 0.9615
         Neg Pred Value : 0.5000
              Precision : 0.9615
                 Recall : 0.9615
                     F1 : 0.9615
             Prevalence : 0.9286
         Detection Rate : 0.8929
   Detection Prevalence : 0.9286
      Balanced Accuracy : 0.7308

       'Positive' Class : 0
```

Fig. 4. Accuracy of Linear Regression

```
mse(team.test$pct,predict_1)

## [1] 0.007277765
```

Fig. 5. MSE of Linear Regression

From the result above, we can infer our linear regression model is pretty good, the R^2 is 0.7495, close to 1;the accuracy is 0.9286,and the MSE for test dataset is 0.007277765.

According to the small MSE value for our test dataset, we can know our model is not overfit.

C. Logistic regression

Because the logistic predictive model is more interpretable compared with data mining tools,so we use logistic regression to see the performance of the models.

```
mse(team.test$pct,predict_fit_2)

## [1] 0.00712028
```

Fig. 6. MSE of Logistic Regression

```
Confusion Matrix and Statistics

          true
prediction  0  1
         0 50  2
         1  2  2

               Accuracy : 0.9286
                 95% CI : (0.8271, 0.9802)
    No Information Rate : 0.9286
    P-Value [Acc > NIR] : 0.6289

                  Kappa : 0.4615
 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.9615
            Specificity : 0.5000
         Pos Pred Value : 0.9615
         Neg Pred Value : 0.5000
              Precision : 0.9615
                 Recall : 0.9615
                     F1 : 0.9615
             Prevalence : 0.9286
         Detection Rate : 0.8929
   Detection Prevalence : 0.9286
      Balanced Accuracy : 0.7308

       'Positive' Class : 0
```

Fig. 7. Accuracy of Linear Regression

From the result above, we can infer our logistic regression model is pretty good,the accuracy is 0.9286, which is the same with linear regression model; and the MSE for test dataset is 0.00712028, which is a slightly lower than linear regression model.

Of course, according to the small MSE value for our test dataset, we can know our model is not overfit.

But there is the other thing we have no idea how to explain: we can not find any significant features.

```
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -2.239e+01  4.241e+01  -0.528    0.598
## OffPoss                -3.097e-04  3.762e-04  -0.823    0.410
## Fg2Pct                 -5.544e-01  5.444e+01  -0.010    0.992
## Fg3Pct                  1.217e+01  1.286e+02   0.095    0.925
## NonHeaveFg3Pct         -7.806e+00  1.263e+02  -0.062    0.951
## FtsMade                 3.110e-02  1.786e-01   0.174    0.862
## PtsAssisted2s           8.310e-02  5.657e-01   0.147    0.883
## PtsUnassisted2s        -1.563e-02  5.532e-01  -0.028    0.977
## PtsAssisted3s          -5.171e-02  4.283e-01  -0.121    0.904
## PtsUnassisted3s         8.455e-02  4.551e-01   0.186    0.853
## Assisted2sPct          -2.695e+00  5.740e+01  -0.047    0.963
## Assisted3sPct           4.449e+00  1.181e+01   0.377    0.706
## FG3APct                 6.532e+00  5.123e+01   0.128    0.899
## ShotQualityAvg          2.775e+00  2.182e+01   0.127    0.899
## TsPct                   1.909e+01  5.817e+01   0.328    0.743
## PtsPutbacks            -1.407e-01  2.953e-01  -0.476    0.634
## Fg2aBlocked             5.013e-01  2.867e+00   0.175    0.861
## FG2APctBlocked         -3.875e+01  1.764e+02  -0.220    0.826
## Fg3aBlocked            -1.446e+00  1.035e+01  -0.140    0.889
## FG3APctBlocked          3.366e+01  2.348e+02   0.143    0.886
## AtRimAssists           -1.001e-01  3.148e-01  -0.318    0.750
## ShortMidRangeAssists   -1.153e-01  3.696e-01  -0.312    0.755
## Corner3Assists          1.635e-01  5.112e-01   0.320    0.749
## Def2ptReboundPct        6.702e+00  2.882e+01   0.233    0.816
## Def3ptReboundPct        1.370e+01  1.346e+02   0.102    0.919
## OffFTReboundPct        -8.161e-01  4.239e+00  -0.193    0.847
## Off2ptReboundPct        1.603e+01  4.056e+01   0.395    0.693
```

Fig. 8. All features of Logistic Regression

D. Lasso Regression

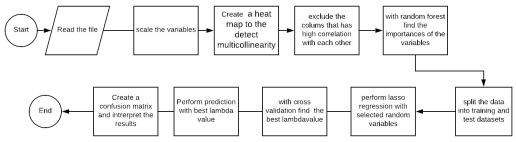

Fig. 9. An Overview of the Algorithm

Since we have over 100 variables and we are dealing with multi collinearity Lasso might be a good fit for the dataset. As can be seen from the flow chart after scaling the variables and detecting the variables with collinearity based on the heatmap and some of the variables are eliminated. A random forest is performed to get the most important variables.

10 variables are selected based on the bar chart above which can be summarized as ppg, fg2pct, fg3pct, opp_ppg, tspct etc.

From the result above, we can infer our lasso regression model is performing with a good accuracy and precision ,the accuracy is 0.91; and the MSE for test dataset approximately is 0.001, which is better than linear and lasso regression.

V. Player Clustering

As we all known, compared with games from early 2000s, the games today are more fast paced and more 3 points focused.

Not only has the analytics era of the NBA dramatically reshaped shot selection across the league, but shooting is by far the most important component of winning games. Teams with a higher effective field goal percentage (eFG%) than their opponents won 81 percent of their games during the

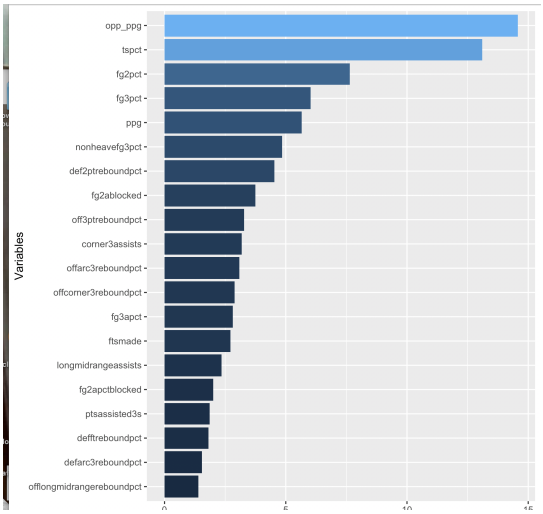

Fig. 10. Random Forest Variable Importances

```
> mse
[1] 0.001359386
```

Fig. 11. MSE of Lasso Regression

```
prediction   0   1
         0 130  12
         1  13 145

               Accuracy : 0.9167
                 95% CI : (0.8794, 0.9453)
    No Information Rate : 0.5233
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8329

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9091
            Specificity : 0.9236
         Pos Pred Value : 0.9155
```

Fig. 12. Lasso Regression Results

regular season, and they're winning 90 percent of them in the playoffs[8].

A traditional NBA lineup consists of 5 positions: *Center, Power Forward, Small Forward, Shooting Guard, Point Guard*. Recently, people start revisiting this definition, and coming up with new ideas of how to define a modern NBA player. Muthu Alagappan [9] declares that current 5 position criterion doesn't really tell us the true role a player plays in the lineup. For example, Shane Battier and Lebron James are all Small Forwards, but what those 2 does on the court are drastically different.

Inspired by the work above, we tries to use the statistics we collected of all (except "outliers") players in the last 10 seasons to find the different *clusters* among all the players.

A. Methods and Steps

Clustering refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a dataset. We seek to find an optimal scenario where within each cluster, the observations are quite similar to each other, whereas observations in different groups are quite different from each other.

1) K-Means Clustering: K-Means Clustering tries to make sure the similarity within each cluster by minimizing the total within clusters variation.[6]

2) Linear Discriminant Analysis: Linear Discriminant Analysis (LDA) is a method that finds a linear combination of features that characterizes or separates two or more classes of objects or events. It uses *Bayes' Theorem* to assign an observation to the class where the *Posterior Probability* is the maximum among all the possible classes. The *Prior Probability* denotes the probability that a randomly chosen observation belongs to a certain class. It also states that observations are drawn from a multivariate Gaussian distribution $N(\mu_k, \Sigma)$, where μ_k is a class-specific mean vector, and Σ is a covariance matrix that is common to all K classes.

3) Principal Components Regression: Principal components analysis (PCA) is a popular approach for deriving a low-dimensional set of features from a large set of variables [6]. The main idea of PCA is to find the directions among all the data provided along which the observations vary the most.

4) PCA+LDA: One of the difficulties by applying LDA directly to the data set is that we have a quite large number of predictors in hand (over 50, even after removing the multicoliearity we still have over 35 features). This is a quite large number of dimension given the dataset size we have. (Around 5000 rows), and will suffer from the *Curse of Dimensionality*. To reduce the number of dimension, we propose to use PCA to do *Dimension Reduction*, then use the principal components as input predictors to do LDA.

To make sure the shrinkage of number of predictors still have a good prediction performance, we choose the number of components from PCA by comparing the prediction accuracy in *Cross Validation* with the prediction accuracy with normal LDA using all the available predictors.

After running the experiments, we decided to use 18 principal components and running the LDA on those 18 components. The accuracy for classify a player to his real position is around 0.74.

5) Choose The K value: There are many methods helping decide what is the appropriate number of K is. In this report, we choose to use *Gap Satistic* as the approach to deciding the optimal K number [10]. Gap statistic is a goodness of clustering measure, where for each hypothetical number of clusters k, it compares two functions: log of within-cluster sum of squares (wss) with its expectation under the null reference distribution of the data. In essence, it standardizes wss. It chooses the value where the log(wss) is the farthest below the reference curve.

To summarize the workflow, we first use PCA+LDA as the measure of deciding the number of principal components to

be used for K means clustering, then by implementing the Gap statistic, we get the number of K we will be using.

B. Results and Findings

We divided the dataset into 2 periods, period 1 is from $2009-2013$ seasons, period 2 is from $2014-2018$ seasons. The K number for period 1 K_1 by Gap Statistic is 7, and the K number for period 2 K_2 is 6. The results for period 1 are shown in Fig. 13 and Fig. 14. The results for period 2 are shown in

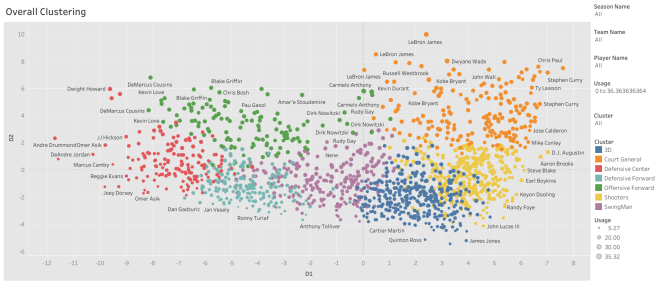

Fig. 13. Clustering Result from 2009-2013

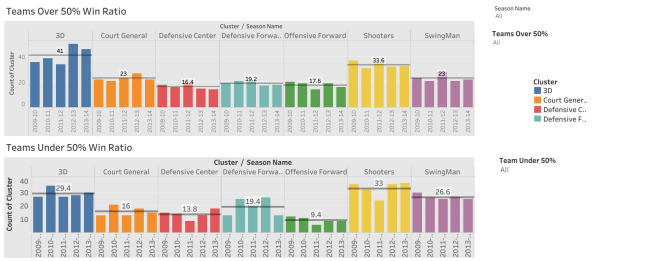

Fig. 14. Good Teams VS Bad Teams from 2009-2013

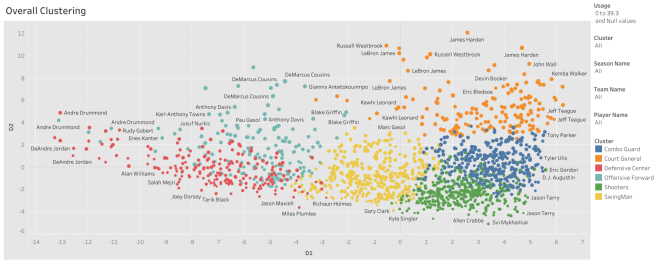

Fig. 15. Clustering Result from 2014-2018

In Fig. 14 and Fig. 16, we define “good teams" as those who were above 50% win ratios in individual seasons. “bad teams" were those whose win ratios were below 50%. Comparing Fig. 13 and Fig. 14 with Fig. 15 and 16, we could notice some interesting difference between those 2 period:

- 3D players and Defensive Forwards are disappearing from this league in modern NBA. (at least not many since they are not a cluster anymore.)
- More and more Combo Guards are taking the stage.
- In period 1, good teams had more 3D players (e.g. Travor Ariza in that early 2010s' Lakers Team)

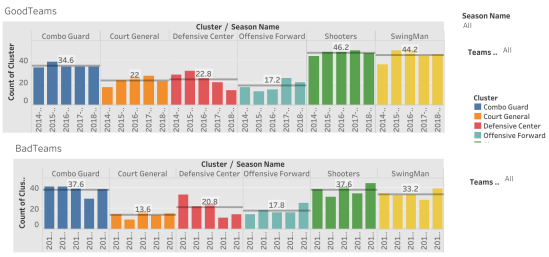

Fig. 16. Good Teams VS Bad Teams from 2014-2018

- In period 2, good teams tend to have more Shooters (Thanks to Golden State) and Swing Man.

For explanation purposes, the Court Generals are refering to the leader or top players of the game, 3D players are used to describe those players who mainly focus on defense and shooting 3 pointers. Combo Guards is the mixture of running the offense and giving out assists. Swing Man are refering to those whose positions are very flexible and will be able to fit for multiple positions.

Besides that, we also noticed for some players, their roles in the team have been constantly changing over the years. For example, in Fig. 17, Andre Iguodala was the obsolute leader back when he was in Philadelphia 76ers and Denver Nuggets. But after signing with Golden State Warriors, he started to take a less “ball dominant" role and finally became a Shooter during the Warriors title run.

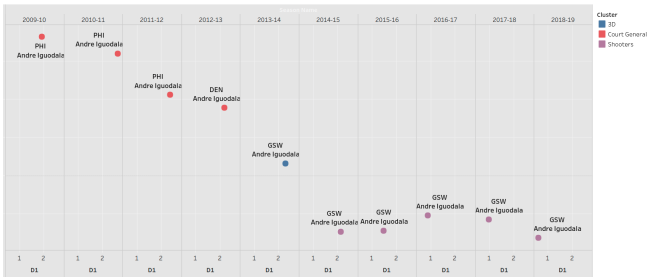

Fig. 17. Andre Iguodala's roles over the years

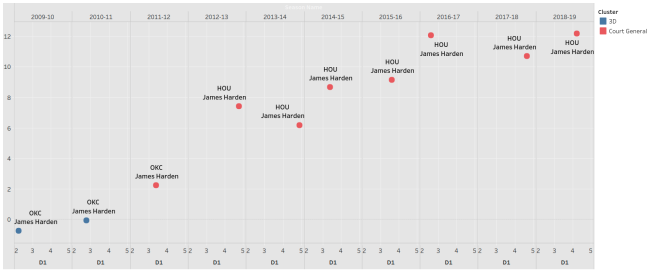

Fig. 18. James Harden's roles over the years

Another big Star in today's league, James Harden, was taking a 3D role back in his first couple of years in Oklahoma City Thunder, and took a big leap after getting traded to Houston Rockets, as shown in Fig 18.

VI. Conclusion

In the previous team analysis section, every metric evaluation or validation result show that based on the features we selected, both linear regression and logistic regression can get a better fit.

From the linear regression model, it seems OffPoss is the most important features, it has the most significant p-value. However, for logistic regression model, although the model can get a high accuracy, there is no significant feature. We guess this is the real life dataset, we can not get the result like the book, there are so many unpredictable factors.

Also, with player clustering, we found a plenty of interesting trends and changes that are resonating with the realities.

$*$ The code and database information could be found at our git repo. The visualizations can be found here.

References

[1] John Hollinger. Calculating per. `www.basketball-reference.com/about/per.html`.

[2] Official NBA Website. Stats home. `https://stats.nba.com/`.

[3] Basketball Reference. `www.basketball-reference.com/`.

[4] PBP Stats. `www.pbpstats.com/`.

[5] ESPN. `www.espn.com/nba/`.

[6] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[7] Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[8] Kirk Goldsberry. *SprawlBall: A Visual Tour of the New Era of the NBA*. Houghton Mifflin Harcourt, 2019.

[9] Muthu Alagappan. From 5 to 13: Redefining the positions in basketball. *MIT sloan Sports analytics conference*.

[10] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a dataset via the gap statistic. 63:411–423, 2000.