University of Padua

Department of Mathematics "Tullio Levi-Civita"

Master Thesis in Computer Science

NBA players and their categorization. An analysis using Data Mining and Machine Learning techniques

Supervisor
Prof. Annamaria Guolo
University of Padua

Master Candidate
Francesco Penna

Academic Year
2022-2023

"When performance is measured, performance improves. When performance is measured and reported back, the rate of improvement accelerates."
— Pearson's Law

Abstract

This thesis undertakes a comprehensive analysis of NBA players' offensive categorization utilizing Data Mining and Machine Learning techniques. This analysis is based on a dataset marked by the absence of advanced offensive metrics, particularly player tracking data. Despite this limitation, the study, spanning four regular NBA seasons and including data from over 2000 players, establishes the viability of its scopes.
The investigation reveals that distinct classifiers successfully address specific tweaks of the categorization challenge. K-means clustering proves useful at discerning broad player categories, while Principal Component Analysis (PCA) avoids overfitting of the data. Notably, the study uncovers the inherent limitations of the dataset in capturing intricate offensive behavoirs from players, which, to be fully uncovered, would require to use both a wider and more complex dataset.
In dissecting offensive impact of each of the discovered clusters, also called *categories* of players, the research employs Principal Component Regression (PCR) and Ridge Regression, revealing their comparable efficacy. This insight suggests that, despite the inherent complexities of offensive metrics, these regression models offer consistent performances.
The significance of this research lies in its novel role within the context of a relatively sparse literature on the subject of NBA players categorization analysis. The study's use of a partially complete dataset serves as a starting point for further exploration and refinement. By revealing the nuances of player categorization and offensive impact, this work establishes a foundation for future research. It is a matter of fact that additional inquiries and methodological advancements in the evolving landscape of NBA player analysis would be helpful on various aspects, from the team building done by the NBA franchises, to scouts and coaches.

Contents

Listing of figures

Listing of tables

Listing of acronyms

..... POS: Position
..... GP: Games Played
..... MPG: Minutes Per Game
..... MIN%: Minutes Percentage
..... USG%: Usage Percentage
..... TO%: TurnOver Percentage
..... FTA: Free Throws Attempts
..... FT%: Free Throws Percentage
..... 2PA: 2 Points Attempts
..... 2P%: 2 Points Percentage
..... 3PA: 3 Points Attempts
..... 3P%: 3 Points Percentage
..... eFG%: effective Field Goal Percentage
..... PPG: Points Per Game
..... RPG: Rebounds Per Game
..... TRB%: Total Rebounds Percentage
..... APG: Assists Per Game
..... AST%: Assists Percentage
..... BPG: Blocks Per Game
..... TOPG: TurnOver Per Game
..... SPG: Steals Per Game
..... VI: Versatility Index
..... ORTG: Offensive Rating
..... DRTG: Defensive Rating

1 Introduction

This chapter serves as an introduction to the work of this thesis. It introduces basketball and some of its fundamental concepts to inexperienced readers on this matter, so to facilitate the understanding. We then talk about the scientific problem at hand, its applications in the world of NBA basketball, and the tools used for the analysis. Finally, we will introduce the structure of this document.

1.1 Basketball

Invented in 1891 by the founding father of the sport, James Naismith, basketball is a team sport, played by five players. Two teams play against each other, and win by realizing more points than the opponents. Points are generated by throwing the ball in the basket, or rim, of the opponents: depending on the position from which this happens, a shot can count for 2 or 3 points. Basketball is, by his nature, divided in two main components for each of the two teams:

- Offense: when the team possesses the ball, and has to score.
- Defense: when the opposing team possesses the ball, and they have to stop them from scoring.

This research will focus on the offensive side of the game. A technical reason for this choice can be found in the main metrics by which players are measured when playing: while there are lots of statistics regarding the offensive end of the floor, the same cannot be said for the defensive one. Also, offense is the main reason for which basketball is watched by so many people. Fans want to see their teams score, in a fun and efficient way, meaning there will always be a particular attention for the offensive production of a team. This does not discredit the defensive side of

basketball, which is equally interesting, and deserves attention as well as a proper research. At the same time, basketball is a worldwide spread sport, but we still decided to analyze data from a specific country. The data from our study are gathered from the National Basketball Association (NBA), which is the most famous basketball league in the world. One of the reasons for this choice relies, yet again, in advantages about data. The NBA has been studied and analyzed by now for years, and data are collected about the players for each game, and each individual season, since the 1970s. Additionally, while preparing for the 2013-2014 regular season, the league also started collecting *tracking data* about players and teams. These refer to even more advanced metrics and statistics on the offensive production for a player, such as the types of *play* a player run in order to score. Another reason is, even in this case, related to popular interest and demand: although in the US television shares went down during the last years, the NBA is still by far the most followed professional basketball league in the world. This brings to the league lots of money and capital investments, which in practice translates to a higher interest in maximizing the performances of the players in a team.

We want to try to delve more deeply in the sport of basketball, so to introduce the actual work of this thesis. Being more than a century old, basketball has seen many changes and different play styles over the years. And while it is beyond the scopes of this thesis trying to synthesize and coherently explain each of the past development of this sport, we still need to give some information in order to better understand what we want to achieve. Giving the low number of players at any time, and the small field in which they have to move in (28 meters, which become 14 meters when a team goes to offense, since they can use only the offensive half when they attack), since the dawn of the sport coaches tried to give each player a role. Also, being a sport in which a player is required to reach an elevated surface, the rim, taller players are more suited for the game. This does not imply that short people cannot play the game, but it pushes even further the need for a clear distinction in the roles, or positions, that each player should keep at any time. Traditionally, there are three macro positions, which are then detailed into five. We decided to stick to the official NBA description of each of these role, as they are given in the official website. These are of course simplified descriptions, but serve the purpose of understanding what each player is expected to do in each position.

- Guards
  - Point guards: runs the offense and usually is the team's best dribbler and passer.
  - Shooting guards: usually the team's best shooter. The shooting guard can make shots from long distance and also is a good dribbler.

- Forwards
  - Small forwards: plays against small and large players. They roam all over on the court. Small forwards can score from long shots and close ones.
  - Power forwards: does many of the things a center does, playing near the basket while rebounding and defending taller players. But power forwards also take longer shots than centers.

- Centers: usually the tallest player on each team, playing near the basket. On offense, the center tries to score on close shots and rebound.

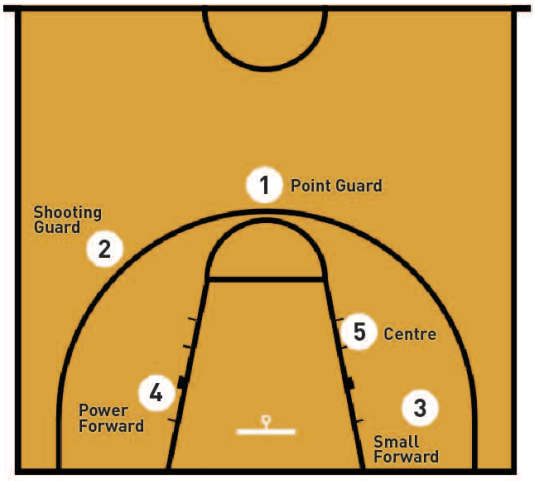


Figure 1.1: Traditional basketball positions in an offensive scenario, from Kevin Bonsor, *How Basketball Works: Who's Who.*

What is seen in Figure 1.1 has been, since the founding of the NBA in 1946, the go-to positions in basketball. Each kid who wanted to start playing the game have been assigned one of these, and each coach run, to some extent, plays and schemes depending on these definitions. Although they are not monolithic, they are the *de facto* standard.
These definitions shaped the most effective ways to score points over the years. Knowing them, we can elect three styles of play which, starting from the 1960s, up to the modern day, defined the best and most effective strategies to score in a basketball game. Recall yet again that, for the scope of this introduction, these are simplified distinctions, which cannot consider, for means of space and time needed, all tweaks and currents NBA basketball saw over the years.

- Big man era (1960s-1990s): being the position which play the closest to the rim, the center can shoot really easy, and high precision shots, often called layups. Being such an efficient strategy, coaches, during the early days of the league, decided to dish the ball as often as possible to the big man in their team, which was, most times, the center. It is not a case that the most influential players in this span of times were all centers. Bill Russel, Wilt Chambairlain, Kareem Abdul-Jabbar, and a long series of players whose main role was to stick close to the basket and take safe shots, ones which was very probable to generate 2 points.
- Mid range era (1990s-2010s): when Micheal Jordan came into the NBA in 1984, it was not a clear revolution the idea of shooting from the midrange, the position of the field further from the rim, but inside the three point line. There were, in the previous years, players which were famous in their own right while sticking to the inner side of the painted area, but no one had, or have been able to reproduce, the effectiveness of Micheal Jordan.

His influence, given by the fact that he was elected the most valuable player in the league (MVP) five times in the 1990s, cannot be underestimated, and shows a clear break with the past. Players such as Kobe Bryiant, Manu Ginobili, Dwayne Wayde, can be seen as the heirs of this style of play. It was now preferred to stick further to the rim, in order to avoid big man inside the area.

- Three point revolution (2010s-today): while for Jordan we noted that there was, even before him, a tendency of shooting from inside the area, what happened starting from 2009 is unprecedented. In the 2010s, teams started took a big increase in amount of three point shots, due to two main reasons. The fact that they are worth more points than two point shots, and the presence of the Golden State Warriors, and in particular the player Steph Curry, in the NBA. The latter played an important role in showing to the world, together with his coach Steve Kerr, what a great number of three point shots per game can bring in terms of points scored. It is less effective than layups and midrange shots, but they are worth more. And in a general scenario, the more attempts a team does, the more it will succeed in three point shots. This phenomenon is regarded today as the three point revolution, and is guided by players as Steph Curry, Klay Thompson and James Harden.

We can gather a main conclusion from this very brief history of the NBA: over the years, less importance was given to the position of a player, and more attention was put on the play-style. This change in mentality is what created the idea for this research, which we can now proceed to analyze.

1.2 Objectives of the research

From Section 1.1 we were able to gather a simple, yet insightful conclusion. In the modern NBA, we see a switch in the paradigm and in the attentions of coaches and general managers alike. Positions, as we described them, and as they are presented nowadays, are less and less useful. Modern players, the ones each team would want to build around, are the ones who are either good at many things, or extremely good in one fundamental style.
But this is not revolutionary in its own as a concept. For years now scouting staffs have evolved in order to meet these new needs. We are used now to see NBA personnel watching everything, from college basketball games, up to European championship ones. And this cannot be seen in any other way than as an effort in order to create a new generation of modern players. Our research, in this sense, is born from a need: studying and analyzing a player from the inside out of his game can be extremely complex, and require lots of hours of work. We want to facilitate this job, by creating an Artificial Intelligence (AI) model which, starting from the statistics of a player, is able to define its *category*. Then, for each of the generated categories, we want to understand which is the best way of the players belonging to it to influence the offensive outcome of a game, through some Data Mining techniques. A category, with respect to a role, can be much more flexible. It can associate players who are similar in their style of play based on what they actually do on the floor, which is measured through statistics. It is easy to see now why preferring the NBA was a logical choice for such a project. Basic statistics about players have been registered for years, and

hence there is the access to a potentially large dataset of players for an AI model. And although some data, referring to before the advent of the internet, may still be a bit complex to access, the ones from latest years are more widespread.
Such a tool would benefit many components of a NBA franchise. Starting from the scouts, described earlier, going all the way up to managers and coaches: it would help introducing new players into a specific team, by finding, for example, the best fit based on the current needs. Straying away from the NBA world, this thesis can also be used in realms such as sports statistic, which is most times associated with betting and sports predictions, more in general. Knowing in advance which are the specialties and tendencies of a player on the offensive end of the field could help drastically in formulating predictions.

1.3 Structure of the thesis

This chapter only serves as an entry point to analyze the aims of this thesis. The rest of the work for this document is structured as follows.

- Dataset, chapter 2: describes the dataset used in this thesis, the way it was collected, and the pre-processing that was applied to it. It also serves as a preliminary analysis for the rest of the study.
- Machine learning methods, chapter 3: contains a brief introduction to machine learning and introduces on a theoretical standpoint the methods used in the analysis.
- Machine learning predictions, chapter 4: shows the results of the analysis performed using the methods described in Chapter 3.
- Data Mining methods, chapter 5: contains a brief introduction to data mining and introduces on a theoretical standpoint the methods used in the analysis.
- Data Mining predictions, chapter 6: shows the results of the analysis performed using the methods described in Chapter 5.
- Conclusions, chapter 7: sums up the content of all the results obtained in the work, possible flaws and future scopes of this thesis.

2 Dataset

This chapter introduces the preliminar analysis that is required in order to approach the objective of this work, being it to find, in first instance, Machine Learning models that are able to distinguish players based on their "category" rather than their role. Hence, what we want to produce with this preliminary work is a knowledge of trends in the modern NBA. The dataset we use for this mean is a collection of all individual players statistics in the last four regular seasons. The regular season of the NBA is a period of more than half a year, in which each team in the league plays 82 games against all the others. The statistics referring to the regular seasons are hence computed based on all the games that a player participated in. In the last four years, more than 2500 players starred in the NBA, and this is the foundation of our dataset. Each player gets labeled with his name, and the year for which the statline is referred. For example, four different entries will be present for Stephen Curry. This is done to monitoring the evolution of a player during the years. The seasons took in account are: 2021-2022, 2020-2021, 2019-2020, 2018-2019.
We now take a look at the metrics with which the players are measured, which can be seen as the covariates of our dataset.

- POS: classical position, or role, for a player (guard, forward, center).
- GP: games played in a season.
- MPG: minutes per game.
- MIN%: minutes percentage. Referrers to the percentage of minutes a player participates in his team's games.
- USG%: usage rate. Estimate of the percentage of a team's play that go through a player.
- TO%: turnover rate. A turnover is when a player loses the ball while he is guiding an offensive possession. Turnover rate specifies how many turnovers a player generates every 100 possess.

- FTA: free throw attempts. When a player is fouled while he is shooting, he gets to throw two or three uncontested shots, known as free throws. They score one point each.
- FT%: free throw percentage. Percentage of accuracy on free throw shots.
- 2PA: two point shots attempts.
- 2P%: two point shots percentage. Percentage of accuracy on two point shots.
- 3PA: three point shots attempts.
- 3P%: three point shots percentage. Percentage of accuracy on three point shots.
- eFG%: effective shooting percentage. If we count as an attempt from ground each shot, without considering the type and excluding free throws, we can obtain FGM (from ground makes), and FGA (from ground attempts). Then, effective shooting percentage is computed as FGM+(213PM). This is FGA computed in order to account the bias of 3 point shots being less accurate but worth more.
- PPG: points per game.
- RPG: rebounds per game. A rebound is counted when a player, either during offense or defense, collects a loose ball after it has been shot.
- TRB%: total rebound percentage. An estimate of how many rebounds of a team are took by a player.
- APG: assists per game. An assist is counted when a player passes the ball to a teammate who manages to score.
- AST%: assists percentage. An estimated percentage of teammate field goals a player assisted while on the floor.
- BPG: blocks per game. A block is counted when a player stops an opponent from scoring.
- TOPG: turnovers per game. 6
- SPG: steals per game. A steal is counted when a player manages to grab the ball from the opponent.
- VI: versatility index. Versatility index is a metric that measures a player's ability to produce in points, assists, and rebounds.
- ORTG: number of points produced by a player per 100 total individual possessions.
- DRTG: number of points allowed by a player per 100 possessions he individually faced while staying on the court.

We considered these metrics to be sufficient for two main reasons. They are the simplest to gather, since they are widespread and easy to access from many sources. Also, we considered them to be complex enough to give a clear distinction between categories of players. Using more thorough metrics could be helpful in highlighting some interesting trends, but it would require access to a much more difficult to obtain set of data. The data used for this analysis were obtained from NBAStuffer, a commonly used platform for gathering data about sports in the US. Their terms of service does not specify issues while using the data that are free for download on their site, but a set of them is available only on payment. The NBA official website, the most obvious choice,

displays all the data we needed for this analysis, but they are not directly available for download. And, although there are some solutions such as *Python* plugins able to download those data, the terms of service specify that a download of any sort is not accepted under any circumstance. Finally, the four CSV files obtained from NBAStuffer where merged together thanks to a *Python* script. The rest of the analysis is instead done with the *R programming language*[3] (R Core Team, 2023), due to the presence of standard functions, as well as many libraries, specialised in statistical analysis of large datasets.
From this point onward we begin the preliminary analysis of this set of data. The way we will proceed is the following, and will be the same for each "traditional" position.

1. Analyze the trends that, in the recent years, have emerged for a determined position. For example, the explosion of three point shots for guards, or the increment of driving plays for forwards.

2. For each trend, define a set of "centers" of this clusters. Players who can be considered the elite for that particular specialty and are hence elected as the standard to follow. Each of these trend is considered a category of players.

3. Check how much the categories overlap. For example, if we find out that two categories share the exact same players, we consider them to be weak, while independent ones are considered as strong starting points for the classification model.

This way, we are able to highlight the differences and variety inside the traditional positions, which is the starting point to move towards a satisfying AI model.
The first step that is required in the analysis of our dataset is, as imaginable, the act of importing it. This gives us also the opportunity to look for different problems or inconsistencies in the data. We are, in particular, searching for:

- Missing data in one or more variables.

- Factor variables that might need to be correctly read by the software.

As expected, the transformation from CSV to R dataframe, the type of object used by the programming language, creates an inconsistency with the factor variables. There are only two in our dataset, that is the team of a player, and its position. We convert those by hand into factors. Done this, the only thing we have to consider now are the NA's regarding ORTG and DRTG. These are, in particular, some advanced metrics that are computed using a large variety of stats and data of a player, defensively and offensively. For the scope of this research, we are not interested in training a model with players who did not play enough. It would imply adding to the dataset a section of players who do not have meaningful information on offensive performances. Hence, we decided to delete them from the dataset. As a consequence, we lose roughly 100 players. These are mainly rookies, players who are in their first professional year, G-League players, who are sent to a minor league to get better, and bench players, which are not meaningful for our purposes (while still deserving all our respects for being professional athletes of the NBA).

2.1 Traditional Roles

This section gives the reader a graphical representation of the current state of "traditional" positions, and how they are used in the league as of today. These labels are pretty stale and not expressive enough, as we believe. The dataset we employ utilize a hybrid categorization with the positions. It includes the three main roles, guards, forwards and centers, but expand them not in the way we showed in the introduction. Rather they create the following hybrid positions, G-F, F-G, F-C, C-F. These are supposed to include players who, for example, have played both as guard and forward, or forward and center. The order of the letters then tells us which is the position the player was implied the most. While distinguishing players by position, these players can be included in either the categories they belong to.
The first plot we want to display is hence how much usage percentage each position has received in the last four years. The variable USG% is, in this sense, a common metric in the NBA, used to give an estimate of how many team plays, in percentage, employ a specific player. More specifically, we are looking at how much players who play in a specific role are used for making plays on the floor.

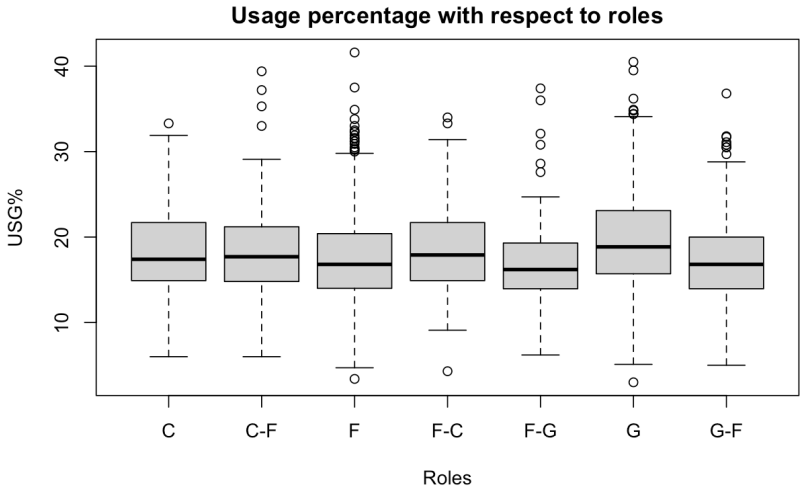


Figure 2.1: Boxplots representing the USG% of players with respect to their role.

Figure 2.1 shows that, overall, there is a uniformity in the distribution of the roles and the usage percentage. This implies that almost each role is used with the same frequency by teams during their games. We will not see, for example, centers stay on the floor and not contributing to the game. Some exceptions are related to two main positions.

- G: guards are a complex group, containing both shooting guards and playmakers, and in the last 10 years have seen bigger and bigger usage. This is the natural consequence of the three point shot revolution, guided by Stephen Curry.

- F-C: nowadays, to accommodate a better shooting percentage, classical Centers (high, heavy and stationary) are replaced by more complex figures, which move in a hybrid way. In this sense, there has been an increase in centers and big forwards who are able to create game by assisting teammates and shoot from long distance. We will investigate this in our work.

A good way to see how much positions are influential in the modern NBA is also to analyze how many players fall in a determined role. For example, seeing too many players in a single position would imply that the considered position is not expressive enough.

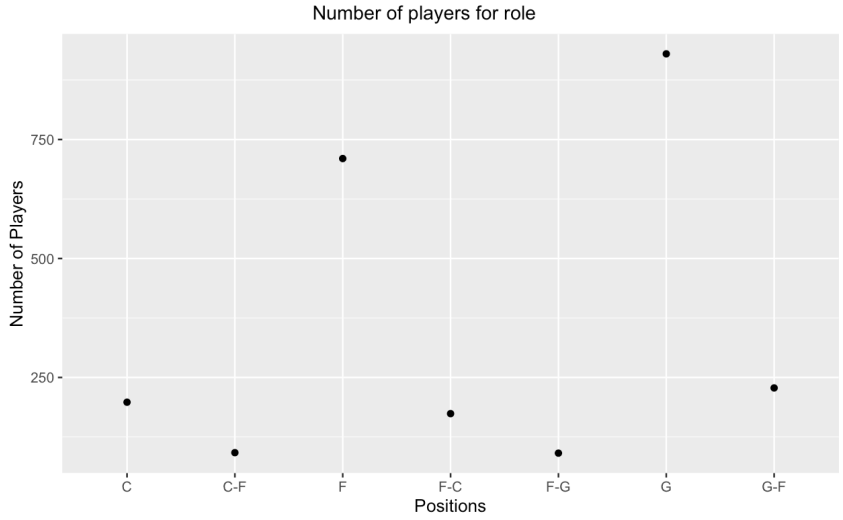


Figure 2.2: Scatterplot representing the number of players that, over four regular seasons, played in each position.

What we assumed above is clearly the case, as Figure 2.2 highlights. Guards alone are not distinguishable this way, since 930 players cannot play in the same manner. Forwards also see a similar problem. Even in the case we allow a distinction, between point guards and shooting guards, and power forwards small forwards, these roles are not manageable. A coach would not be able to infer interesting data from these distinction alone, and would resort to watch tapes, himself or with the help of staff, to better understand the novelties of the game of a player. Being the most crowded position by far, we considered interesting to start directly from them.

2.2 Guards

On a higher level, which does not include statistical measures related to tracking, we can define three main roles that guards, whether modern or traditional in their style of play, have to excel at in the NBA.

- Three points shooting. The heritage of the last 10 years of the NBA, it is impossible, as of right now, to imagine a guard which has not a solid three point shot in his arsenal. If that's the case, we want to examine in which ways the guard can still become dangerous in producing score.
- Generating points through assists. The so called "ground generals" are a more traditional idea of guards, but they cannot be considered outdated. Considering the latest trend in three point shooting, an assist is now worth around 2.4 points. This is because finding an open teammate who can shoot without a defender upfront is still the most efficient way to score.
- Two points shooting. While being still less worth than a three point shot, it is a fundamental which cannot be overlooked. Two point shots can still be reliable, and guards tend to prefer them in what is called "isolation", or off a screen. Isolation can be defined as a play in which one offensive player has the ball with other offensive players nearby, but not in close enough proximity to where the ball handler can pass. Instead, shooting off a screen happens when a bigger sized player blocks the defender of the guard, creating a so called "screen", which allows to take an uncontested shot.

On a last point before proceeding, we wanted to note a detail about two point shooting for guards. While it would be possible to create an even further distinction between what are called "hero players", which takes lots of isolation posses, and players who utilize well screens, we are limited by the expressivness of the dataset. These information are contained in what are called "tracking statistics", which the NBA started officially registering in 2013. Due to their age, and the fact that they can be extremely complex to gather, there is still not a reliable, open source way to gather them. Hence we will stick, for the moment, to the category of two point shooting guard.

2.2.1 Three point shooting

We can define elite three point shooters in the modern NBA players that shoot from behind the arch with two conditions.

- Shoot more than 5 threes per game. On an 82 games per season, 410 threes overall.
- Shoot with a precision of 35%, meaning at least a third of the shots will generate points.

Players as such, which are an increasing category, can be defined as the elite shooters from three points in the NBA. Obviously not each player specialized in 3 point shooting can be considered an elite specialist. In particular, the green points we see in Figure 2.3 should be considered as centers of a natural cluster related to players who are specialized in 3 point shooting.

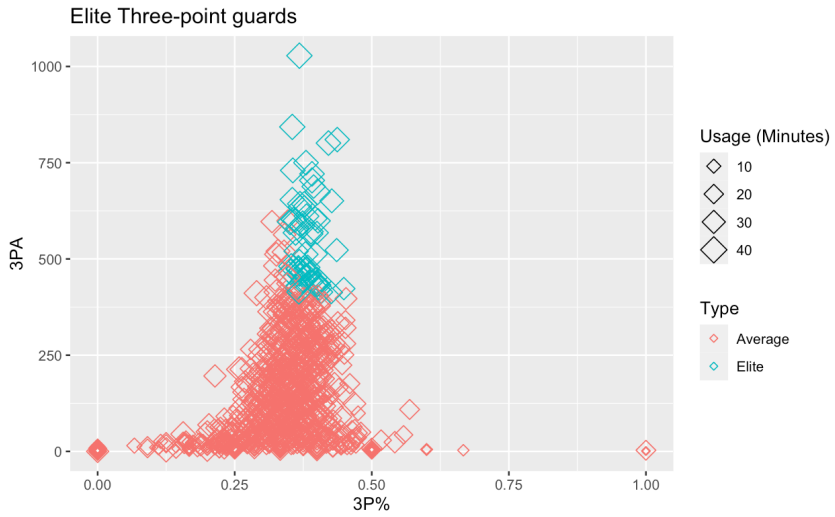


Figure 2.3: A plot showing the normal distribution of 3 point shooting for players considered Guards. Blue points represent elite shooters.

2.2.2 Ground generals

Similarly to what we have done for three point shooting, we have to define conditions to understand what a ground general is. A guard as such must satisfy the following requirements.

- Provide at least 6 assists per game.
- Have an AST% of at least 35%. This would imply that at least a third of the assists provided by the player are of good quality, reaching a free teammate for an easy shot.

Players as such are a valuable asset for a team, since they posses what is called, in the field, "basketball IQ". They are the players who can set up complex plays and free teammates for an easy shot to the basket. Hence, while being really strict requirements, there are still 80 players who qualify for this category, which are represented by the blue points in Figure 2.4.

2.2.3 Efficiency landscape offensively for guards

Even with the few data we have gathered so far, it can be interesting to assess a possible flaw our methodology for the research. In particular, we want to see if there is a style of play which is particularly better than the other in terms of efficiency impact on the scoring of the game. Let's put for example that a three point shot is able to generate three point even only 20% of the times, and a guard start to takes lots of them. Doing this, would imply that a classification model could correlate a high index of three point attempts to a high offensive rating for a player. This would then lead to classify into a three point specialist even a player who cannot be considered as such. To prove that these categories are equally, or so, efficient, we provide Figure 2.5, that shows how ORTG is impacted by three point shooting and assists.

Figure 2.4: A plot showing the distribution of assists for players considered Guards. Blue points represent ground generals.

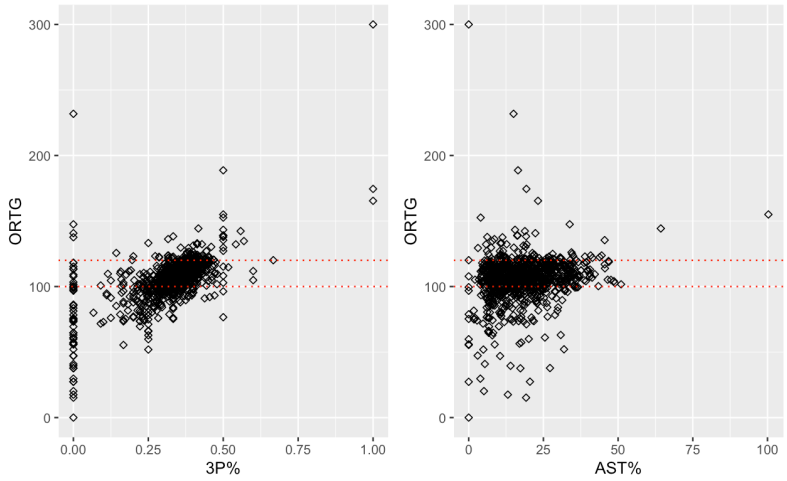


Figure 2.5: Scatter plots used to analyze ORTG with respect to 3P% and AST%.

Comparing these two plots allows us to gather insightful relationships. The clouds of points are similar and follow a recognizable pattern. This implies that shooting well from 3 point and providing good assists are both good means for a guard to impact on the scoring. A consequence should be in the fact that we should not see each and every guard mapped to only one of these two specialties. The interval highlighted in the plot serves the point of showing a "good" ORTG, which is between 100 and 120 points for 100 possesses. Another interesting result is in understanding what can be a good mean of 3P% to be considered a three point specialist. We see that a consistent ORTG is reached when a player shoots between 25% and 50%.

2.3 2-POINT SHOOTING

Going even further with respect to guards we arrive to the 2 point shooting. We purposely did not include this analysis in the guards section, due to the nature of this kind of fundamental. The two points, in all their forms, are the most widely used mean of improving the score for each and every type of position.

- Guards may utilize their small frames in order to penetrate the area and shoot from midrange.
- Centers generally utilize their bodies in order to "post-up" defenders, ending up with fade always or layups. Going to post is the practice that implies a player attacking while being turned from the defender, in order to defend the ball. It allows, if the player is able to find a way, to score points directly under the rim.
- Forwards may utilize both these tools. This implies that a complete analysis of the players who behave well on two-point shooting must consider each and every position. We want to see if particular roles have a bigger tendency in generating score with 2-points, as well as discover whether there is a distinction inside the roles themselves.

This implies that a complete analysis of the players who behave well on two-point shooting must consider each and every position. We want to see if particular roles have a bigger tendency in generating score with 2-points, as well as discover whether there is a distinction inside the roles themselves.

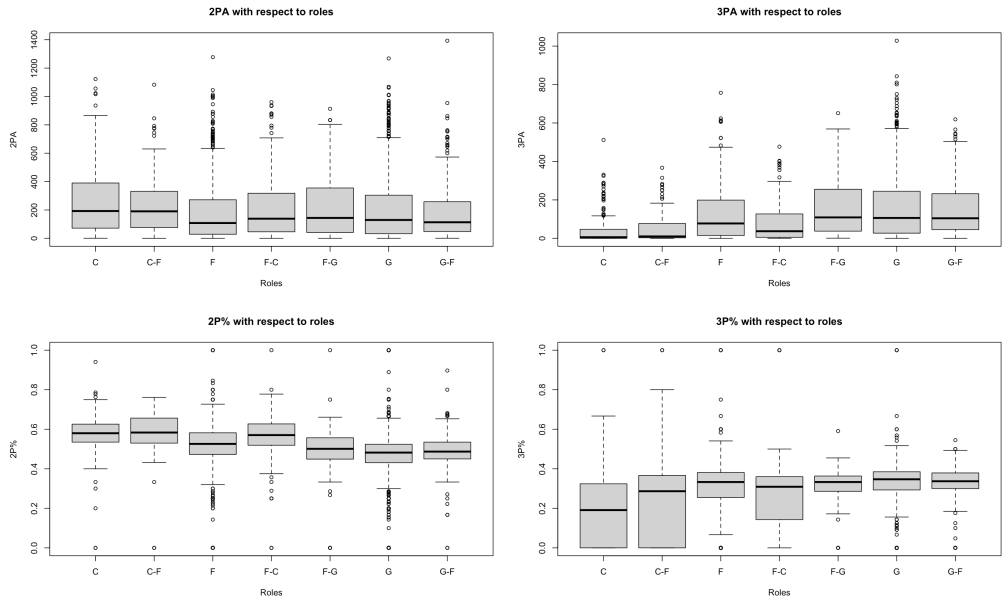


Figure 2.6: Boxplots showing shooting efficiency of shooting per role.

Figure 2.6 allow us to gather some first interesting information about 2-point shooting, with respect

to the 3-point case. In terms of attempts alone, all roles more or less try the same quantity of shots from inside the arch. This is a first result that suffragettes our initial hypothesis: all roles equally prefer 2-point shots, meaning there it is suffocating to have roles so limiting. Instead, as far as 3-point shots go, guards are the premier terminals for such weapons. Together with forwards, and their hybrid categories, they are the roles that tend to take most of these shots. Interesting is the clear presence of outliers in centers, since it hints towards a rising category of players, commonly known as "modern centers".
As far as the quality of these two type of shots goes, it is not a surprise to see that 3 point shots will have lower chances of going in: they are performed further from the rim. In a similar fashion, it is easy to see that centers, forwards, and their hybrid roles, will tend to have a higher percentage on 2 point transformations, since they tend to position around the rim and in the paint. Finally, guards are the category which suffers the most the 2 point shoot: we see a big issue in relating under-performance, since lots of shots have bad quality. This can be related to the presence of "deep twos", which are 2 point shots made far from the rim, almost on the 3 point line. Overall, we see that half of the 2 point shots are transformed by each role, since the mean of 2P% in our dataset is 0.5049. But since we described briefly how different these type of shots may be, we want now to analyze perks and differences. First of all, how determinant is the 2 point shot in terms of scoring efficiency for a player? We can gather this information from a simple scatter plot about 2 point accuracy and points per game of the players.

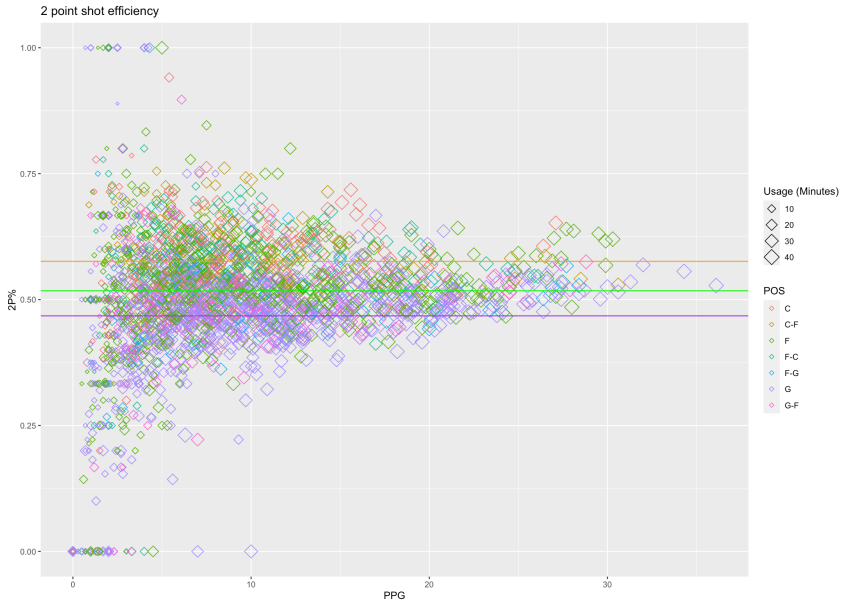


Figure 2.7: 2P shooting efficiency in the whole NBA.

Results of Figure 2.7 are particularly interesting. Again we see an overall mean in the two point shot transformation which is almost exactly $\frac{1}{2}$. This exemplifies why so much players will tend to take an overwhelming amount of these shots. That being said, some data we wanted to highlight

are about the relationships between pure roles and the two point shot.

- Guards, being the most overcrowded position by far, are also the position which tend to take the most of these shots. But most interestingly, we see that their efficiency is below the mean of the NBA as a whole. Guards indeed tend to shoot 0.4677022% from the two point area, as it was already seen. This can be connected to the difficulty of the shots taken by guards, as well as to the overall preference to go for open 3 point shots, as seen in the previous section. That being said, we will see in a moment if there is a category of guards who indeed prefer the 2 point shot with respect to the 3 point one.
- Forwards, being a sort of hybrid terminal of attack, in between centers and guards, are the position which sticks the most to the mean of the NBA on the two point shot. There are mainly two ways in which a forward is expected to score from two points. Midrange shots are the ones taken from inside the area, and are a classic basketball shot to the rim. Layups are instead the act of penetrating the area with an athletic effort, and finishing close to the rim, for higher efficiency. The evolution of the game in the last years has backed off from the midrange to behind the three point line, meaning the overall shot preferred by this category is the layup. We will see later whether there are players who still prefer the midrange as a tool to score.
- Centers, being the players closest to the rim, will tend to have easier shots, and indeed shoot with 0.5759899% from the two point area. But, as we have seen, there are also centers who will nowadays prefer to alter the three point shot with the two point one, meaning we can hypothesize a division inside this position.

2.3.1 Guards and 2 point shooting

Firstly, we start by analyzing the efficiency landscape of two point shooting among guards, as we did for the three point shot, and the results are shown in the Figure 2.8 scatterplot.
As for before, we can define an elite two point shooter as a player averaging the following statistics:

- Shoot more than 6 two point shots per game. 500 overall in an 82 games season.
- Shoot with a precision of 50% or more, which, as established, is the current average of the NBA, and higher than the average for the role itself.

As an interesting result, we can analyze that this cluster is not as populated as expected. The evolution of the game deeply impacted what a guard should nowadays do, and indeed there is a set of "just" 50 players, which is still much less with respect to the previous categories.

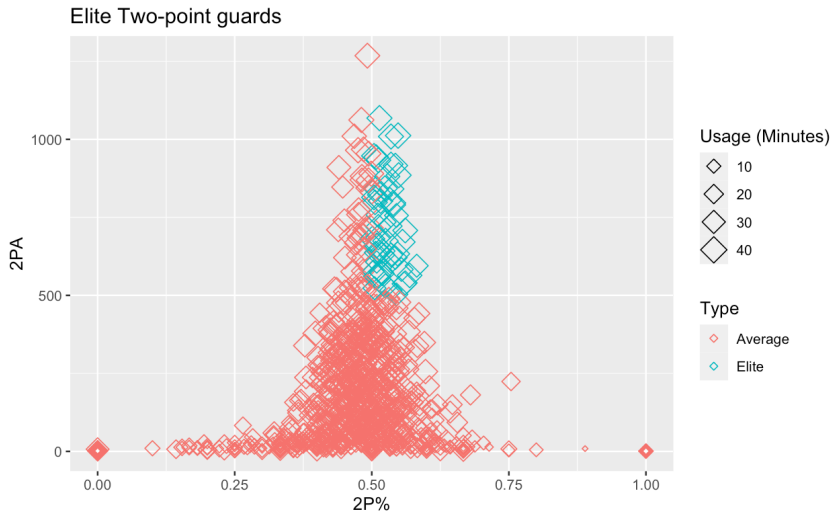


Figure 2.8: A plot showing the normal distribution of 2 point shooting for players considered Guards. Blue points represent elite shooters.

2.3.2 Considerations on guard's clusters

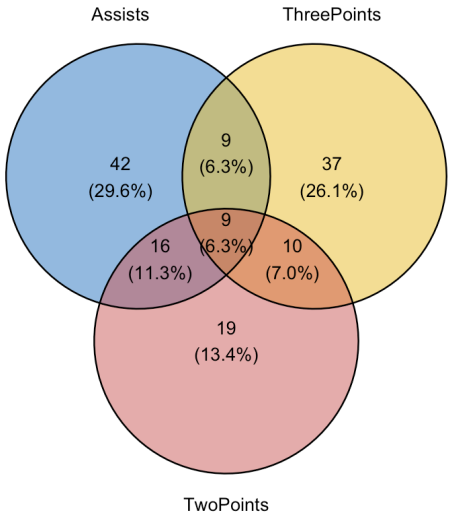


Figure 2.9: Venn's diagram to analyze the intersections between the player's categories analyzed for guards.

Before moving forwards to other positions, we wanted to give a last consideration on guards. What we intended to do in the first place was to find interesting categories, and then comparing the sets of players found, to see how much they overlap. We show this result through a Venn's diagram, in Figure 2.9.

What we see from this diagram is surely encouraging. Recalling that these are supposed to be ideal clusters, we see an independence between different clusters. It can be seen, fist of all, in how much the pure assisting guards and three point shooting guards are independent from one another. The only intersection which gets a considerable amount of players is the one between two point shooters and assists specialists. This is due to a more traditional style of play, and should not come as much of a big surprise.
Overall, the results prove to be satisfying, and can help us lead the way in the next section, in which we proceed to analyze forwards.

2.4 Forwards

Similar analysis to the ones we have seen for guards could be done here for forwards. It can be tricky indeed to identify which are the specialties of such an hybrid position. They should be good enough to compete with small and big framed defenders, meaning their expertise should be wide overall. That being said, for the sake of diversity, we want now to try and see other metrics, which could help us identify particular types of forwards. In particular, we will be looking for the following.

- Driving forwards. As hinted before, driving is the act with which a forwards penetrates the area and goes close to the rim, to get a high percentage shot. A possible flaw in studying this metric is in the fact that we do not have access to tracking statistics, and it is hence complicated to study such a behavior. The solution we came up with is to analyze free throws. The idea is simple: the more a player will try penetrations, the more he will get fouled, since it is very easy to be touched during a shot while running towards the rim. This cluster will be hence identified through free throw percentage.
- Three point shooting. Although we already did a similar analysis for guards, one of our main interests in this research is to get a cohesive analysis of how much the three point revolution has impacted the modern NBA. Hence, we will try to see how many big framed players, supposed to work inside the area, behave with respect to three point shots.
- Rebounding proficiency. This is one of the more "traditional" aspects of a forward. While not being a center directly under the rim at all times, it is still expected to box out smaller defenders, and take a loose ball when is comes on the floor.

2.4.1 Driving forwards

Due to the difference in the frame of the players, during the years the biggest distinction between guards and forwards were drawn by size and athleticism. In general, the trend is for the last type of players to try shots closer to the rim. A good way to analyze how much of an impact the penetration of a player has on the flow of the game is the free throw percentage.
Figure 2.10 is able to acquire interesting information about the relationship between forwards and free throw attempts. In particular, the dotted line represents the NBA mean in free throw attempts. It shows, on its own, that there are lots of player which can regularly cut the area and go for a drive towards the rim in the role forwards. Then, the elite players are defined as the ones to shoot with 75% from the line, meaning they will sink $\frac{3}{4}$ of their attempts. In this sense, we are able to see a big group of specialists of this style of attack. Indeed, there are 143 players which enroll in this category.
The further requirement, for shooting efficiently from the free throw line, is needed to clear out some players who cannot be otherwise considered efficient in attack, and hence cannot be considered centers of a cluster. Also, we address the issue in analyzing drivings to the basket this way,

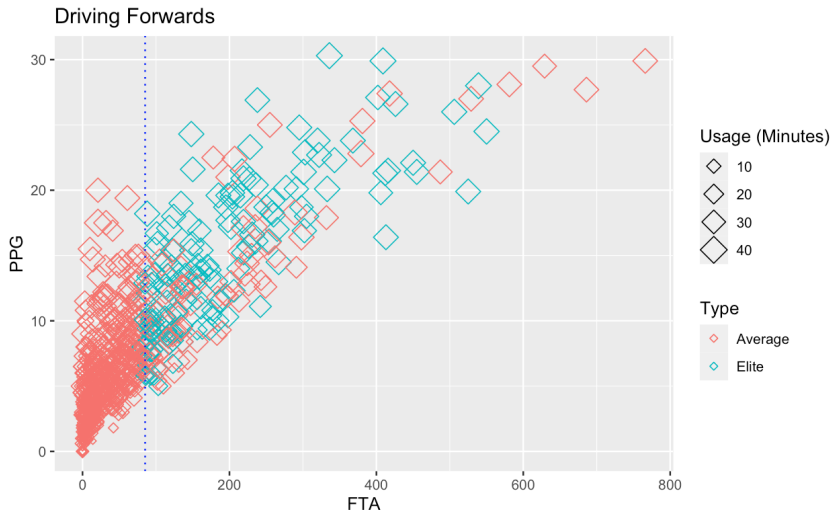


Figure 2.10: A plot showing the distribution of drivers for players considered Forwards. Blue points represent elite drivers.

and recognize is a bit of a logic connection. We still believe that, with the exception of some noise, this is a representative cluster, due to the less and less use of midrange of today players.

2.4.2 3 POINT SHOOTING

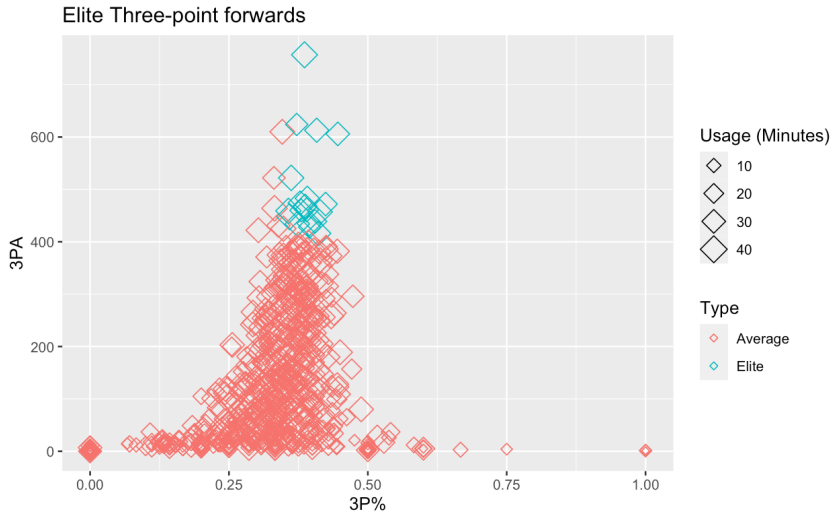


Figure 2.11: A plot showing the normal distribution of 3 point shooting for players considered Forwards. Blue points represent elite shooters.

Figure 2.12: A plot showing the distribution of rebounds for players considered Forwards. Blue points represent elite rebounders.

The result from Figure 2.11 is interesting, not for the distribution seen in the plot, which is similar to the one of guards, with less players of course, due to the nature of a three point shot, but rather in the players which showed up in the results. There are only 19 players in this category, and only one (Duncan Robinson) has been consistent during all the last seasons in being a member of this subset of forwards. Other players, such as Lebron James in 2021/2022, became only recently a good shooter. It shows an interesting trend in which elite shooters can be drawn from many different categories. We chose to study only the case in the pure forwards role, being there more observations. Also, these kind of players tend to be historically pretty different from guards, in terms of size and style of play. This shows hence an interesting melting pot between different positions.

2.4.3 Rebounding proficiency

In order to understand which forwards are considered proficient rebounders we took a precise bottom line:

- The player has to have taken at least 6 rebounds per game.
- The player has to have taken at least 12% in TRB% for the season.

These value are not randomly generated. They are the mean for rebounding metrics in the center position. Being centers considered the players who should take most rebound in a game, we wanted to analyze whether there is a case for forwards to mix in this specialty, and this is indeed the case.
Figure 2.12 shows that there are 68 players who belong to this category, showing a big increase

in the fact that forwards should also belong closer to the rim, and confirming how hybrid this position is in style of play. This, along with the previous result of forwards who tend to shoot from 3 point and forwards who prefer to drive to the rim, shows an interesting trend in the variability of specialties forwards have achieved in the analyzed period.

2.4.4 Considerations on forward's clusters

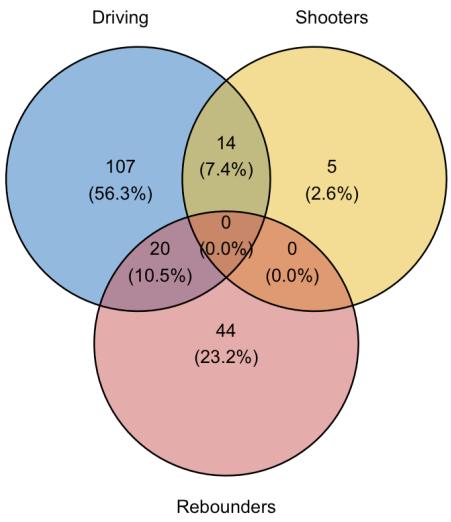


Figure 2.13: Venn's diagram to analyze the intersections between the player's categories analyzed for forwards.

We have so far been able to discover three main clusters for the role of forwards. In particular, they were the driving forwards, the long distance shooters, and the rebounders, which can be seen as the main roles that such a role might perform (in still a very general scenario). We want now to see how much these clusters overlap with one another.
The results, assisted via Figure 2.13, prove to be extremely satisfactory. We see that rebounders and sharpshooters never overlap with one another, proving to be completely separated. Rebounders share with driving forwards some players, but are still well separated (66% of rebounders are explicitly good at this specialty). Finally, shooters share with driving forwards a big part of their cluster, but we have to recall how the latter specialty was computed. It was done by analyzing trips to the free throw line, meaning this data can be influenced by fouls done on the three point line to shooters.

2.5 Centers

So far we went out and did not consider 'hybrid roles' present in the dataset due to a main reason: the guards and forwards role were already really populated with observations. This is not the case for the pure center role, meaning we will consider, in the means of this analysis, also the category Center-Forward. These are players who played in both positions during their career. The same can be said for the hybrid role Forward-Center, hence also this is considered. Doing so, we go from a sample of 198 observations to one of 464.

2.5.1 Classic centers

Basketball, as we will see time and time again in this research, has evolved a lot across the years. In the early days of the game, there was a pretty defined tendency, which gave the name to the mainstream playstyle: the big man era. Big men, in this sense, can be identified as centers. The definition of classic center can guide our research efforts into two main areas, which so far we have already seen for other roles. We want to study rebounding proficiency of centers, trying to find a cluster of efficient rebounders in the modern NBA, as well as a more general category of hyper-efficient two point shooters. The last consideration is linked to what we already have seen in the 2 point shot introduction: being the centers close to the rim, they must take easier shots, and hence we want to find the ones who shoot from inside the paint with pin point precision.

Rebounders

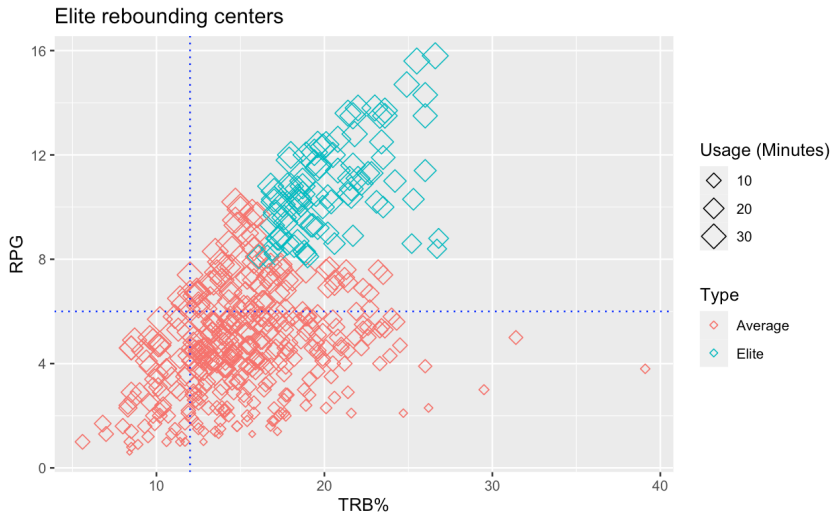


Figure 2.14: A plot showing the distribution of rebounds for players considered Centers. Blue points represent elite rebounders.

By doing a simple query, we could see that the bar set for efficient forwards rebounders is too low. Being it formed by the mean of the center position, it is not a good way of analyzing which are the really efficient ones. This can be seen by the vertical dotted line showing the mean of TRB% for centers, while the horizontal one shows the previously expected 6 rebounds per game. The new conditions we used are considerably higher:

- The center must take 8 rebounds per game.
- The center must be involved into the 16% of rebounds that happen while he is playing.

This are of course much higher expectations. In basketball, when a player achieves more than 10 points and rebounds, he achieves what is called a "double double". This becomes a "triple double" if he manages to create 10 assists too. We are asking basically our centers to average a double double or triple double including rebounds, as well as a participation in almost a fifth of all defensive and offensive rebounds that happen on the court. But even with these high requirements, Figure 2.14, we can see that a lot of centers are up to the task.

Paint specialists

The same analysis we introduced for guards who can be considered skilled from over the arch will be done here for centers, as shown in Figure 2.15. We are able to identify an interesting cluster of size 65, but the most interesting piece of information from this plot can be gathered from the position of the points. In particular, we are able to see in practice what we have analyzed so much up until this point: being closer to the basket centers will take easier shots, and, in practice, have a higher 2P% with respect to other players. This is easy to understand from the fact that we see really few centers that shoot under 50% from 2 points.

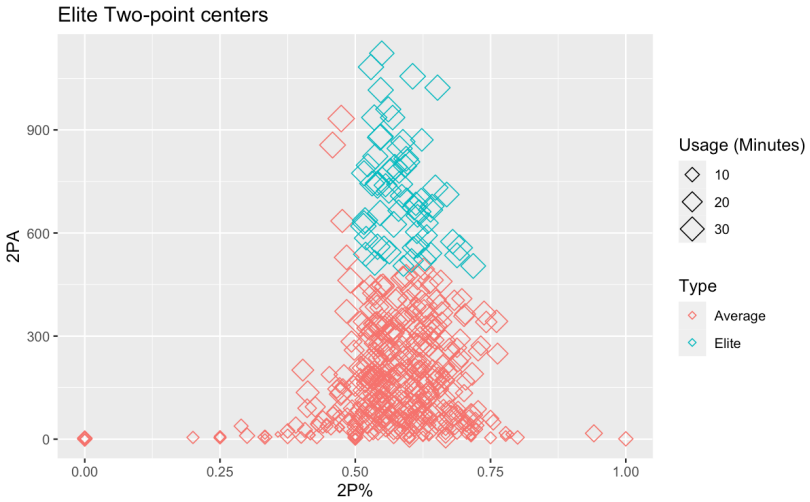


Figure 2.15: A plot showing the distribution of 2 point shooting for players considered Centers. Blue points represent elite shooters.

2.5.2 Modern centers

"You can't teach height". This has been, for many years, the guideline that coaches and general managers in NBA utilized to create good teams. As we said, big man era brought to the league a generation of big centers, which had to utilize size and weight to fight directly under the rim. And the reason for that is simple, as we just saw: close shots under the rim are the most efficient weapon that players has to generate scoring, still to this day. This was true until the introduction of the three point line, in the 1979/1980 season. Of course, it took years in order for teams to adapt to this new introduction, and consequentially step away from the rim, but eventually this happened. Also, due to revolutions in the rules for charges and fouls in the paint, centers lost the opportunity to be "bullies", which was a big style of play. We finally enter the 2011/2012 season, where the league, understanding this trend, opted for a big change: in the all star voting ballots for 2012 the position "center" was removed, and instead it was merged with the forwards, creating the backcourt. This is a real revolution for the NBA, since the following season teams attempted more than 20 threes a game, signaling that the revolution had started and it was now unstoppable. We come this way to the modern day centers: players who have a skillset similar to a playmaker, players who built over the years a respectable three-point shot, and centers that start from outside the area and drive towards the rim. All of these new tendencies open the floor for a more complex analysis, overcoming the original concept of center position. In this sense, we hope now to illustrate meaningful clusters of centers with particular strengths, comparable to guards or on-ball forwards.

3 point shooting

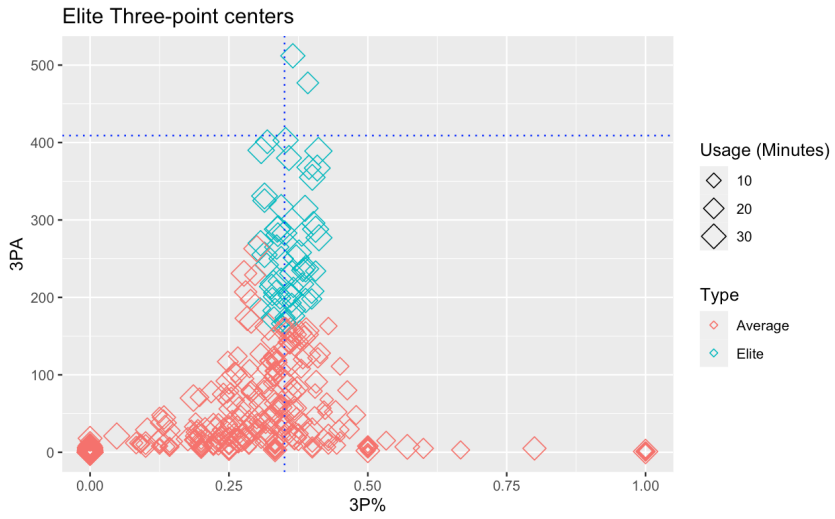


Figure 2.16: A plot showing the normal distribution of 3 point shots for players considered Centers. Blue points represent elite shooters.

We introduced this cluster of players earlier in our search, highlighting that there were some outliers in the 3PA and 3P% boxplots for the center positions. Now we want to see whether or not there is a real case for shooting centers, which would be a huge deal in the modern NBA. Small sized guards could not guard a center that decides to lift his hands and shoot, due to the physical mismatch. Obviously, we cannot keep the requirements we had for 3 point shooting guards to define what a shooting center is. It would be way too restrictive, as the dotted lines in Figure 2.16 highlight. According to the official NBA website the mean of 3PA for centers is of roughly 2 3PA per game, meaning that, in an 82 game season, we will see 164 attempts. Also, we estimate a 30% shot efficiency, which is not a huge drop from the 35% set for guards. This overall builds the profile of a versatile center. The numbers may not seem that much outstanding, but they serve a purpose: imagine being a coach that has to guard a profile as such. What would be the best option when he gets to the three point line? Such a player can drive, pull a jump shot, or pass the ball. Overall, it is not possible to find a player who is ready to guard all these options with exceptions for the best defenders in the league. This simple realization is what makes players as such so much valuable in modern NBA. Another point we would like to address is how, over the year, the presence of shooting centers can be seen as a positive trend. In the league, we want to analyze whether or not there has been an "arms race" towards such profiles.

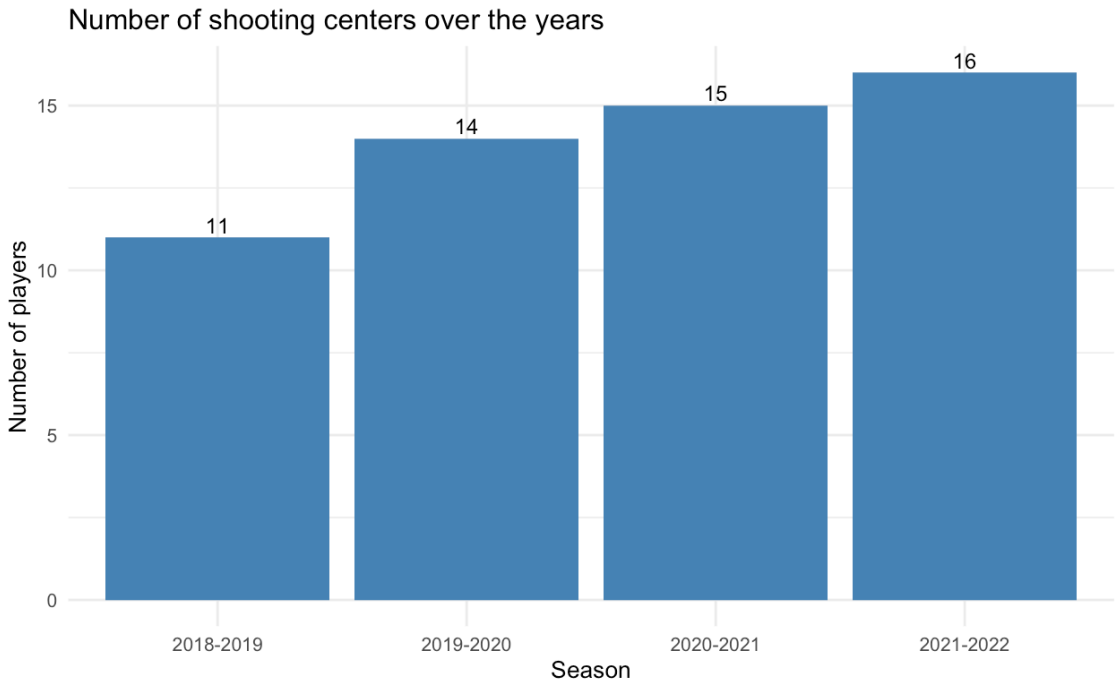


Figure 2.17: Diagram showing the increase in presence, over the years, of shooting centers.

This is an encouraging result for a future research. There has been a steady yet continue growth in the field of shooting centers, and we can expect that, over the years, we will see an even more extreme volume, with an increasing accuracy, in this specialty.

Big framed point guards

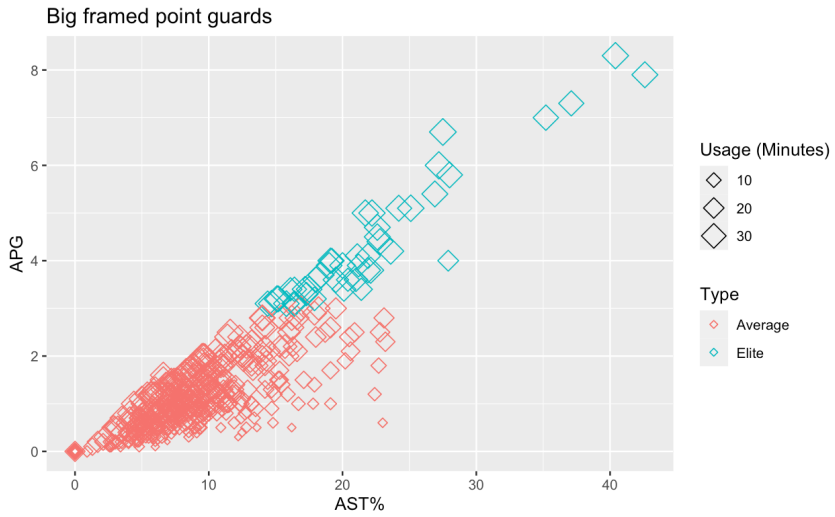


Figure 2.18: A plot showing the distribution of assists for players considered Centers. Blue points represent ground generals.

A similar approach as in the previous section can be drawn for assists. Again, the requirements for playmakers and ground generals are way too strict, and we have to tie them down. In particular, we want to see which are the centers who distribute at least 3 assists or more per game.
Yet again, we are able to find in Figure 2.18 a significant cluster, of size 46. The most interesting results is the AST%: we see that, in general, centers will try to distribute less assists, but they are really meaningful for the team. This can be seen in the fact that such a satisfactory result of 35% AST% is reached so quickly by so many players. It is an interesting metric of how, nowadays, centers are expected to have intelligence and keep an eye on every part of the game.

2.5.3 Considerations on center's clusters

As we did for the other roles, we now want to look at the independence measure of clusters. This is found by seeing how many players belong to different categories. It allows us to see how likely a good classifier will be able to create independent groups. Results show that some clusters are, as expected, more independent than others. The plot can be divided by the colors in two categories: yellow regions show clusters of "modern" centers, while in blue we show the classic ones. In particular, we are able to see in Figure 2.19 that classic centers, while being independent in the rebounding category, are also rather mixed up with one another. But the distinction should be clear enough in order to highlight two different specialties, and a more general "classic center" category, able to do both things good. The most interesting result by far is the one of three point shooting centers. As we have seen, they are a rising category, but the results of the plot are very

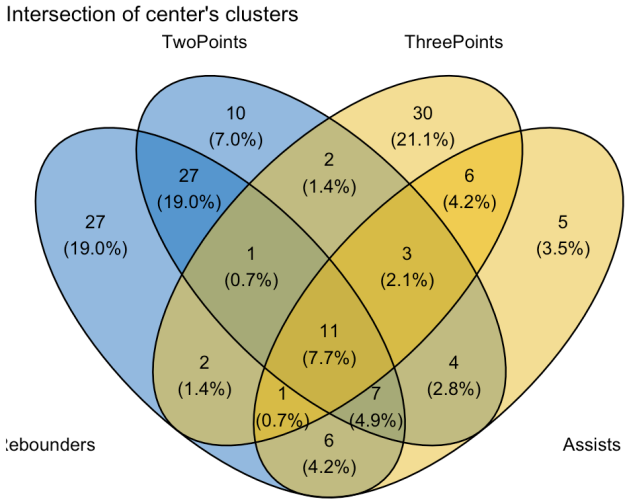


Figure 2.19: Venn's diagram to analyze the intersections between the player's categories analyzed for centers.

encouraging. Indeed this category, in its pureness, is the most populated. A similarly optimistic result cannot be feasible for assist centers as well. The results in its independence are very poor, and it serves the purpose of highlighting that this is indeed an existing trend, but a slowly rising one. The back to back MVP of the league Nikola Jokic is the obvious reference for this style of play, but it might still take a while for this trend to stabilize as a tendency for players and coaches in the league.

2.6 Stars and versatile players

Among all team sports, basketball is the one to have its outcomes dictated by star players the most. This is not a surprise to anyone familiar with the sport: having only five players at any times on the field, in such a small place, will mean that a particularly gifted talent can manually alter the flow of a game. Being the NBA the most important basketball league in the world, it is not a surprise hence that such a number of star players have gathered, over the years, in such a way. Nowadays, the NBA is over the idea of "star power vs team power". Recent proceedings (the player empowerment movement for example) has shown how much a single movement from a star player can alter the balance of all teams in the league. This are the main reasons why we cannot, in our work, ignore the presence of stars. These are, in the modern league, versatile players, able to do pretty much everything above the mean of the other players. Players such as Russel Westbrook, Klay Thompson, Lebron James, Giannis Antetokoumpo and Nikola Jokic have in their arsenal each and every aspect of the game we already studied, and are efficient in both offense and defense. Moreover, each of these players play in a different classic "position", starting from point guard all the way to center.
This is why we will now study the so called "versatility index", an approximate measure which shows how much a player is proficient in different styles of play. An average player scores around 5 in VI, while a star has 10 or over. To study offensive efficiency, we will analyze it in relationship to PPG by a player. Figure 2.20 shows how well the versatility index is positively correlated with the PPG for a player.

Figure 2.20: Scatterplot analyzing Versatility Index and PPG.

Of course, we are talking about the peak of modern NBA, and do not expect to find a big cluster, yet we still want to analyze how well spread it is among the traditional positions, and this can be

seen in the plot from Figure 2.21. The plot is obtained by querying our dataset to find out how many players with $VI \geq 10$ are present for each position.

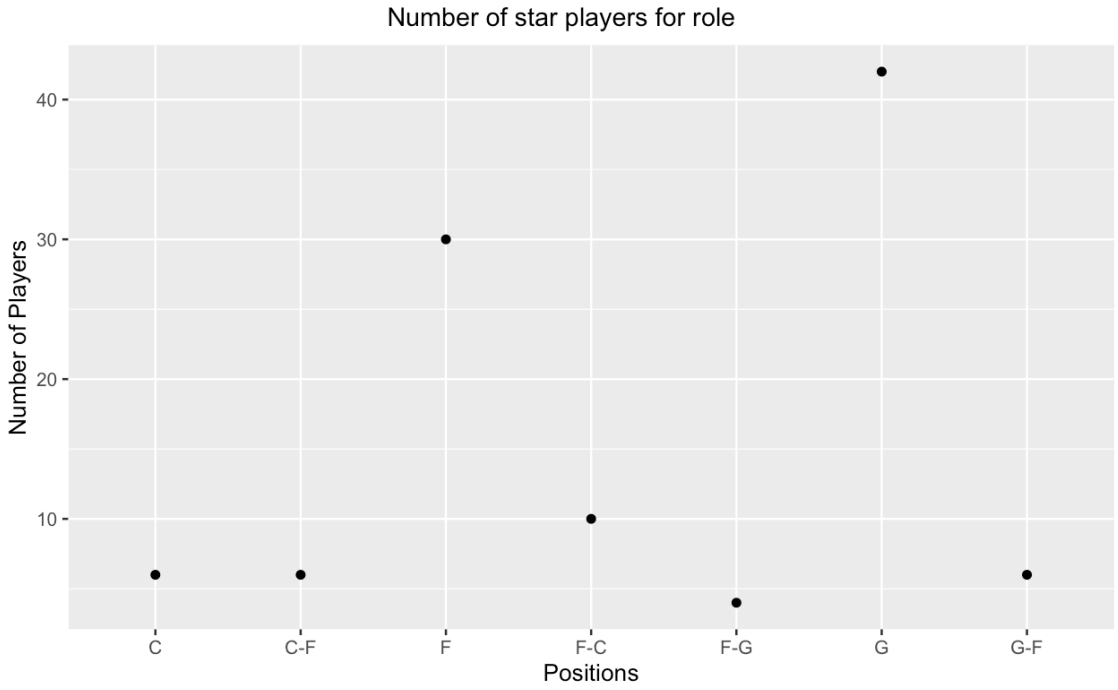


Figure 2.21: Plot showing how many

Results of this analysis are influenced by the size of the original positions. Guards and forwards being so populated with respect to the others will naturally imply the presence of more star players. Still, it is interesting to analyze that guards will tend to achieve more PPG with respect to other categories, showing how much shooting from three points, or creating free shots, can improve the scoring outcome of a game.

3

Machine learning methods

This chapter introduces the machine learning methods to approach the problem addressed by our research. In particular, we are trying to define a model which, starting from a statline of an NBA basketball player, is able to find his "category". With this term, we intend a more precise definition of what the player is good at on the basketball floor, with respect to the current distinction in roles.
The problem we are looking is hence, by definition, a classification one. Starting from a dataset, it is necessary to find an accurate algorithm to create these categories. Despite the presence of studies on classification problems, and a large set of solutions to them, we find ourselves restricted. The dataset introduced in Chapter 2 does not have labels about categories of players. And while this shows how much of a novelty problem this is in the realm of Machine Learning, it leaves us with the necessity to employ a branch of techniques known as *unsupervised machine learning*.

3.1 Framework for unsupervised machine learning

A first high level description of unsupervised learning can be given by describing the differences with its counterpart, supervised learning. In the latter case, a machine is given a sequence of outputs $y_1, y_2, ..., y_n$ and has to learn how to reproduce a "good" output, given a never seen input. In other terms, it goes through a training phase, in which it learns from past data, and then it gets tested to see how well it behaves. Unsupervised learning, instead, receives only inputs as $x_1, x_2, ..., x_n$, but it never gets supervised or rewarded based on how it behaved. It represents, in other terms, the idea of finding patterns in data, going above and beyond noise.

The cornerstones of unsupervised learning are clustering and dimensionality reduction. Both these techniques rely on learning a probabilistic model, starting from the data. The machine is expected to estimate a model that represents the probability distribution for a new input x_t given previous inputs $x_1, ..., x_{t-1}$. In other words, it has to build a learning tool that can model $P(x_t|x_1, ..., x_{t-1})$. This probabilistic model can indeed be used for classification, and find its fundamentals in Bayes rule

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

where P represents the probability, $P(y|x)$ is the probability of class y given the input x, and $P(x|y)$ is the converse. $P(x)$ is the probability of observing input x across all classes.This definition allows to create a statistical framework for machine learning, which in practice influences the techniques we want to address. The learner we are trying to build has beliefs about the world of data analyzed, which have to be translated numerically. By accepting what are known as the *Cox axioms*, it is possible to obtain a remarkable result[4]: if the machine is to represent the strength of its beliefs by real numbers, then the only reasonable way of manipulating these beliefs is to have them satisfy the rules of probability, such as the Bayes rule. This opens up a consequence. Saying $P(X = x|Y = y)$ can be seen as the degree of belief that $X = x$, knowing that $Y = y$.
Hence, the Bayes rule allows to define a simple framework for machine learning, as explained and proved by Ghahramani, Zoubin[5]. Assume a universe of models Ω, where $\Omega = \{1, ..., M\}$, and M is not needed to be finite or countable. The learner will start with some prior beliefs over the models, $m \in \Omega$, such that $\sum_{m=1}^{M} P(M) = 1$. In this sense, a model is simply a probability distribution over data points, in other words $P(m)$. We can assume data to be sampled independently and identically distributed. After observing a dataset $\mathcal{D} = \{x_1, ..., x_N\}$, the beliefs over models are given by

$$P(m|\mathcal{D}) = \frac{P(m)P(\mathcal{D}|m)}{P(D)} \propto P(m) \prod_{n=1}^{N} P(x_n|m),$$

which can be read as the posterior over models is the prior multiplied by the likelihood, normalized. Finally, the predictive distribution over new data is

$$P(x|\mathcal{D}) = \sum_{m=1}^{M} P(x|m)P(m|\mathcal{D})$$

which can be drawn from rules of probability theory, and the fact that the models are assumed to produce independently and identically distributed data.
This latter assumption makes sense with respect to the problem at our hand, as seen in the preliminary data analysis. Data from our dataset do not have a clear structure, meaning the assumption fits well.
Dimensionality reduction and clustering both find their basis in latent variable models, and are the main techniques we will use in order to infer our data. The framework we introduced above is able to fit well a large range of models which include these two strategies.

3.2 Clustering methods

Clustering stands as the most known and studied unsupervised learning problem. It deals with the problem of giving a structure to unlabeled data by creating, as the name suggests, clusters. A cluster is defined, by T. Soni Madhulatha[6], as a collection of objects which are "similar" between them and are "dissimilar" to the ones belonging to other clusters. In this sense, a clustering problem is usually defined by requiring that clusters optimize a given objective function Φ, which satisfies a specific property.
There are two types of clustering.

1. Hierarchical: defined as the set of algorithms which find the successive clusters using previously established ones. These algorithms can further be sub divided into.

    - Bottom-up: begin with each element as a separate cluster and merge them along the different iterations of the algorithms.
    - Top-down: begin with the whole set as an input and divide them along the different iterations of the algorithms.

2. Partitional: defined as the set of algorithms which find all clusters at the same time.

In the most general case, the input of a clustering problem consists of a set of points which belong to a metric space. This is defined as an ordered pair (M, d), where M is a set, and $d(\cdot)$ is a metric on M. A metric can also be called a distance function. The most known is the **Minkowsky distance**, or Euclidean distances, as defined by Jure and Rajaraman (2016)[7].

Definition 1 *Euclidean distance*
Let $X, Y \in \mathbb{R}$, with $X = (x_1, x_2, ..., x_n)^T$ and $Y = (y_1, y_2, ..., y_n)^T$. For $r \geq 1$, the L_r- distance between X and Y, also known as L_r- norm can be defined as:

$$d_{L_r}(X, Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^r \right)^{\frac{1}{r}}$$

This formula, depending on the value assigned to r, can hold different implications.

- $r = 1$. This is referred to as the Manhattan distance. It represents the sum of the absolute differences of coordinates in each dimension, and is useful in grid-like environments.
- $r = 2$. The classic euclidean distance, useful in the $\mathbb{R}^n$ scenario, is also denoted with $|| \cdot ||$.
- $r \to \infty$. It is the maximum absolute differences of the coordinates, over all dimensions.

With this formula it is possible to capture the notion of similarity, which is at the core of algorithms such as K-Means.

Before taking a look at the theoretic aspects of the subject, we must specify that all these algorithms are not designed to provide an optimal solution in polynomial time. This is due to the constraint known as *NP-hardness*. For such problems the best approach revolves around finding an approximate solution, close to the optimal one. Since problems related to clustering can often be represented as minimization or maximization ones, we can hence give the definition of c-approximation.

Definition 2 *C-approximation algorithm*
For $c \geq 1$, a c-approximation algorithm A for a combinatorial optimization problem is an algorithm that, for each instance $i \in I$ returns a feasible solution $A(i) \in S_i$, where S is the set of solutions. Given the objective function for the problem Φ, in case of a minimization:

$$\Phi(A(i)) \leq c \; min_{s \in S_i} \Phi(s)$$

Instead, in case of a maximization:

$$\Phi(A(i)) \geq \frac{1}{c} \; max_{s \in S_i} \Phi(s)$$

The value of c is called approximation ratio, and the instance of the solution $A(i)$ a c-approximation.

3.2.1 Centroid clustering

At its core, centroid clustering can be seen as a subset of center-based clustering methods. The latter are the most known problems in the realm of clustering analysis, including examples such as k-nearest-neighbor or k-center. In practice, a k-clustering center-based, on a set P, is defined as a tuple $C = (C_1, C_2, ..., C_k; c_1, c_2, ..., c_k)^T$, where:

- $(C_1, C_2, ..., C_k)^T$ are partitions of P
- $c_1, c_2, ..., c_k$ are suitably selected centers for the clusters, with $c_i \in C_i \; \forall \; 1 \leq i \leq k$.

With this knowledge, we can now talk about K-means. It is an algorithm which aims at minimizing within-cluster variance. This is done by constructing a minimization problem which finds the minimum possible squared Euclidean distances. Given a center-based clustering, the objective function for k-means can be defined as follows. It is desired a set of centers which minimizes

$$\Phi_{kmeans}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{a \in C_i} \left\{ (d(a, c_i))^2 \right\}.$$

Hence, for each point in the cluster the average euclidean squared distance from its center is minimum. Due to the presence of a quadratic dependence, this can be rather sensitive to outliers. On its own, this problem is of course NP-hard, and hence hard to solve in linear time or space.

The particular property of K-means related to the so called *centroids*, which helps to simplify this problem. Given the notion of Euclidean distance as in Section 3.2, we can define a centroid as follows.

Definition 3 *Centroid*
The centroid of a set P of N points in $\mathbb{R}^D$ is

$$c(P) = \frac{1}{N} \sum_{X \in P} X,$$

where the sum is component-wise.

Note that a centroid is, by definition, not necessarily a point belonging to P. This result is particularly interesting thanks to a Lemma, demonstrated by Jure and Rajaraman[7], which states that the centroid $c(P)$ of a set $P \subseteq \mathbb{R}^D$ is the point of $\mathcal{R}^D$ which minimizes the sum of the square distance to all points of P. The lemma implies that, when seeking a k-clustering for points in an euclidean space which minimizes the objective function Φ_{kmeans}, the best set of centers to select for each cluster is the centroid of each one. This can be done in a variety of ways. During the years, lots of heuristics have been developed which are able to approximate efficiently and accuretely a solution for the k-means problem. Among all, we analyze the three solutions implemented by the standard *kmeans* function in the R programming language R (R Core Team, 2023), used for the analysis in the thesis. We will start from the default choice in R, that's the Hartigan and Wong implementation.

Hartigan-Wong

Developed by Hartigan, J. A. and Wong, M. A.[8], firstly published in 1979, this algorithm is the default choice for many programming languages. The main algorithmic concept behind this solution is in the *local search optimization*. This is utilized by many heuristics used for solving NP-hard problems, and consists in finding a solution which minimizes a criterion among a set of candidate solutions. In our case, we are trying to maximize the objective function Φ_{kmeans}. In the Hartigan-Wong approach, the local search optimization tries to relocate a sample into a different cluster with the intent of improving the objective function. When no sample can be relocated with an improvement, the method stops.

At each iteration of Algorithm 3.1, it is possible to move an observation with two different strategies:

- First-improvement: any improving relocation is applied.
- Best-improvement: each possible relocation is computed and then the best one is applied.

Of course, here the trade-off is relative to speed and precision. The first strategy favorites the efficiency, while the second can get to a closer solution to the global optimum.

Algorithm 3.1 Hartigan Wong Algorithm

Let $\Phi_{kmeans}(S_j)$ be the individual cost of S_j, which is defined by the use of a distance function. Let $(c_1, ..., c_k)$ be the centers of the clusters.

ASSIGNING STEP

Partition the points initially into random clusters $\{S_j\}_{j\in\{1,...,k\}}$.

UPDATE STEP

Given $n, m \in \{1, ..., k\}$ and $x \in S_n$, determine wether the following function reaches a maximum.

$$\Delta(m, n, x) = \Phi_{kmeans}(S_n)+\Phi_{kmeans}(S_m)-\Phi_{kmeans}(S_n \setminus \{x\})-\Phi_{kmeans}(S_m \setminus \cup \{x\})$$

Meaning that, if a point can be moved from cluster S_n to S_m with an improvement of the value of the objective function, its position is changed.

TERMINATION

The algorithm terminates once $\forall\ x, m, n,\ \Delta(x, m, n)$ is less than zero.

An interesting observation is the one appointed by the Authors, where they underline that "it is guaranteed that no cluster will be empty after the initial assignment in the subroutine". Without a condition for termination though, it is possible that the algorithm itself may end trapped in a local optimum, which is a common problem for such heuristics. Many implementations allow indeed to specify a condition to avoid such a danger.

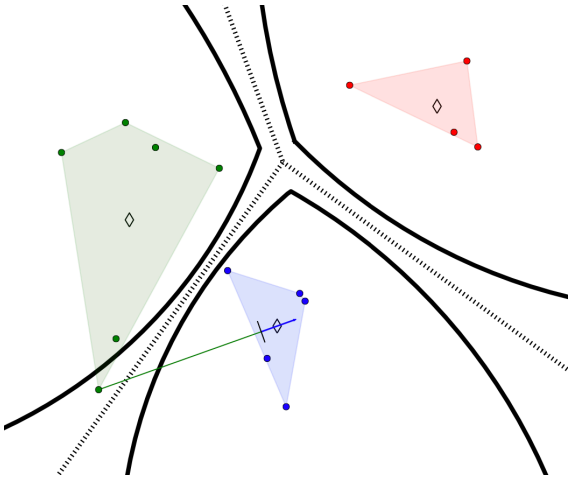


Figure 3.1: Hartigan-Wong algorithm iteration, credits to Matus Telgarsky and Andrea Vattani[1].

Figure 3.1 shows the example of one iteration of the Hartigan-Wong algorithm. Since the hulls represent the dimensions of the clusters, it is easy to see that a green point is too far from its centroid. Based on this, it is reassigned to the blue cluster.

Lloyd-Forgy

Firstly published by Lloyd, S.[9], Lloyd's algorithm is known under many names, such as Forgy's algorithm, or Voronoi iteration. It utilizes to full advantage the theorem we analyzed about centroids. This is possible by repeatedly finding the centroid of every set in the partition, and then re-partitioning the input according to which of these centroid each point is closer.
While being thought for Euclidean planes, and hence a cost function equal to the Minkowski distance with $p = 2$, the algorithm works well even with high-dimensionality and non-Euclidean metrics.

Algorithm 3.2 Lloyd Algorithm

Let $\Phi_{kmeans}(S_j)$ be the individual cost of S_j, which is defined by the use of a distance function. Let $(c_1, ..., c_k)$ be the centers of the clusters.

Assigning Step

Partition the points initially into random clusters $\{S_j\}_{j\in\{1,...,k\}}$.

Update Step

For each partition S_i, $i \in \{1, ..., k\}$, compute the centroid c_i', and create a new partition $\mathcal{C}$. If it holds that:

$$\Phi_{kmeans}(\mathcal{C}) < \Phi_{kmeans}(S)$$

then $\mathcal{C}$ becomes the new partition. The points get assigned to the closest new center.

Termination

When a global minimum for Φ_{kmeans} is reached, the algorithm terminates.

Figure 3.2 gives a graphical representation of this process.

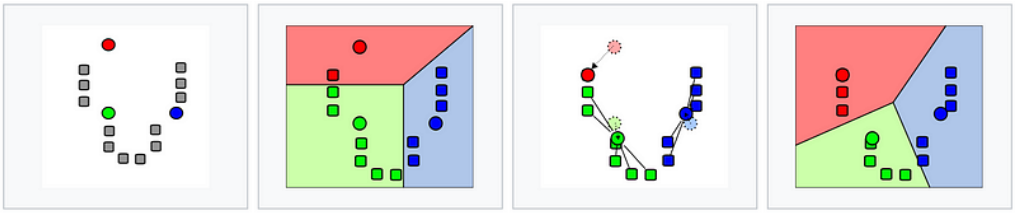


Figure 3.2: Example of Lloyd's algorithm iteration, credits to `https://www.kdnuggets.com/2018/07/clustering-using-k-means-algorithm.html`.

Algorithm 3.2, as proven in[7], always terminates, but with certain limitations. As for the Hartigan-Wong solution though, points are initially randomly split in the partitions, meaning that a local optmimum far from the actual solution may be reached. With $k = 3$ and an Euclidean plane, a

situation such as the one in Figure 3.3 may occur.

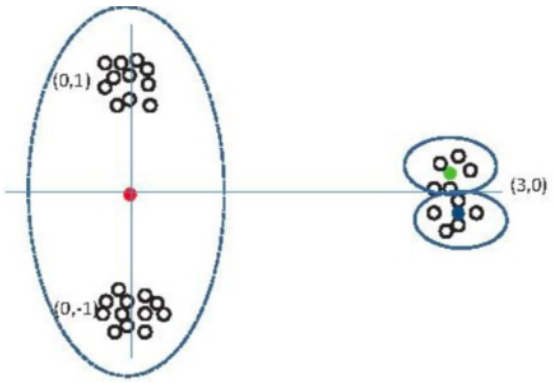


Figure 3.3: Example of Lloyd's algorithm trap.

Also reaching the global optimum, without a condition for termination, may be really expensive on a computational stand-point. On a theoretical standpoint this algorithm is not promising. Nevertheless, empirical studies show that the algorithm requires almost linear iterations to the size of the dataset. This comes as an intuitive conclusion, due to the fact that multiple programming languages offer the support for such an heuristic.

MacQueen

The third and last implemented algorithm for k-means by the *R* programming language is the MacQueen (1967)[10]. It is a simple solution to the k-means problem, that utilizes, as Lloyd's implementation, the concept of centroids and their properties.
When Lloyd's solution tries to update the assignments of datapoints, it does not move the centroids. This is problematic, as with each new assignment the centroid changes its position. It can happen that an observation is wrongfully assigned to a centroid simply because said centroid was not updated. MacQueen tries to cope with this, as it updates the centroids with each new assignment. Obviously this results in additional computational time.
As in the other cases, we are looking at a minimization problem, where we want Φ_{kmeans} to be as small as possible. Again, the distance function can be of any kind, since the algorithm works well in an Euclidean space with multiple dimensions, as well as in other scenarios.
We now proceed to show an high level description of the algorithm, provided by prof. Matteo Matteucci.
The procedure shown by Algorithm 3.3 stays pretty close to the one developed by Lloyd. Being a simple heuristic, or greedy algorithm, it serves the purpose of finding efficiently an approximate solution to this problem. Rather than being a different solution with respect to Lloyd's it tries to

Algorithm 3.3 MacQueen Algorithm

Let $\Phi_{kmeans}(S_j)$ be the individual cost of S_j, which is defined by the use of a distance function $d(\cdot)$. Let $S = (c_1)$ be a randomly chosen centroid in the cluster.

Assigning Step

For each c_i such that $i = 2, ..., k$ assign the next centroid while maximizing $d(c_i, S)$. Assign each point in S to its closest centroid.

Update Step

For each partition $S_i,\ i \in \{1, ..., k\}$, compute the centroid c_i', and create a new partition $\mathcal{C}$. If it holds that:

$$\Phi_{kmeans}(\mathcal{C}) < \Phi_{kmeans}(S)$$

then $\mathcal{C}$ becomes the new partition. The points get assigned to the closest new center.

Termination

When a global minimum for Φ_{kmeans} is reached, the algorithm terminates.

address the biggest weak point of the latter, being the initial position of the centroids. In both implementations, reproducing an experiment multiple times may mitigate this issue.

3.2.2 Density based clustering

An analogy that different clustering methods share resides in the concept of within-group similarity. And although so far we have seen a particularly precise group of centroid-based algorithms, this concept stands still. Density based clustering techniques utilize the same premises already described, a data space with a set of points, and a notion of dissimilarity expressed through a distance metric, whichever the form. Centroid based techniques try to minimize the sum of squared pairwise dissimilarities between cluster objects, starting from a value k expressing the final number of clusters. The results of these assumptions are usually clusters of convex shape.
Differently from what seen so far, density-based clustering is, as for Kriegel, Hans-Peter et al.[11], a nonparametric approach, where all clusters are considered to by high-density areas of density $p(x)$. Being nonparametric, it does not require the number of clusters k as an input parameter, and no assumption is made on the density $p(x)$, or about the variance within the clusters. A direct consequence of this last point is the fact that no clusters are created based on the concept of pairwise within-cluster dissimilarity as measured by a dissimilarity function $d(\cdot)$. Hence, clusters can be of any arbitrary shape, not just convex. The result is, in general, a set of data object spread in the data space over a contiguous region of high density objects. These are separated from other density-based clusters by contiguous regions of low density objects.
The reason to investigate such a methodology is in the concept of "natural clusters". Sometimes density-based clusters can be interpreted this way, due to the absence of the convex shape limitation. The results are hence in particular handy for studies related to nature-inspired applications. Basketball, in this sense, can be seen closer to such a field of study. The simplest implementation to create natural clusters makes use of single-linkage clustering. Given two clusters C_1, C_2, single-linkage clustering measures the distance between them by the following formula

$$D(C_1, C_2) = \min_{x_1 \in C_1, x_2 \in C_2} D(x_1, x_2),$$

meaning that the distance between two clusters is given by the distance between the two closest points in each one. A naive approach to create natural clusters can be defined as following: group all objects below a given distance threshold at a first level, then increase it and repeat this process until all objects belong to a group. This technique has an inherit problem represented by the so called "chaining effect". In this way, different clusters indeed can be connected via the existence of a "chain" of single objects between two clusters. Both Wishart, D.[12] and Hartigan, J.[13] tried to create a generalization of this problem. The latter achieves a more general formalization of a density-based cluster.

Definition 4 *Density based cluster (Hartigan, 1975)*
Given a density $p(x)$ at each point x, a density treshold λ, and links specified for some pair of objects, a density-contour cluster at level λ is defined as a maximally connected set of points x_i such that $p(x_i) > \lambda$.

Definition 4, as well as the one from Hartigan, utilizes the intuition of what constitutes a cluster.

The basic assumption is that the data set $\mathcal{D} \subset \mathbb{R}^d$ is a sample from some unknown probability density $p(x)$, and clusters are high-density areas of this density $p(x)$. Finding such high-density areas usually requires two actions.

- A local density estimate at each point. Typically an algorithm for k-nearest neighbor can be used to address the issue.
- A notion of connectivity between objects. Typically, points are considered connected if they are within a certain distance ϵ from one another.

Clusters are then considered as sets of observations that are connected to other observations whose density exceeds some treshold λ. The set $\{x|p(x) > \lambda\}$ of all high-density objects is called the *density level set* of p at λ. Different density-based algorithms may differ in how the density is computed, how the notion of connectivity is defined, and wheter the algorithm used to detect connected components is scalable or not.

DBSCAN: Density Based Spatial Clustering of Applications with Noise

Single-linkage algorithms such the ones developed by Wishart and Hartigan are pretty naive, and not efficient. When large datasets are added, it is necessary to consider scalable solutions. Density Based Spatial Clustering of Applications with Noise[14] can solve this issue, since it allows the use of index structures for density estimations. We proceed now to explain the main concepts and structures behind this algorithm.
Consider a set of points $\mathcal{D} \subset \mathbb{R}^d$ that needs to be clustered. Let ϵ be a parameter specifying the radius of a neighborhood, applied to any point. DBSCAN clustering classify points in three ways, as follow.

- Core point: a point p is considered a core point if at least *minPts* points are within distance ϵ of it, including p.
- Directly reachable: a point q is considered directly reachable from p if point q is within distance ϵ from point p. Points directly reachable are considered as such if and only if are reachable from core points.
- Reachable: a point q is considered reachable if there is a path $p_1, ..., p_n$, with $p_1 = p$ and $p_n = q$, where each p_{i+1} is directly reachable from p_i. This has the consequence that all points in the path have to be core points, with the only exception of q.

All points not reachable from any other point are *outlier*, also called noise points. If p is a core point, then it forms a *cluster* together with all points reachable from it. Each cluster contains hence at least one core point, and non core ones can belong to it, but they form an *edge*, since they cannot be used to reach more points. All these concepts are shown in Figure 3.4.

For the example, let $minPts = 4$. Red points, among with the one labeled as A, are core

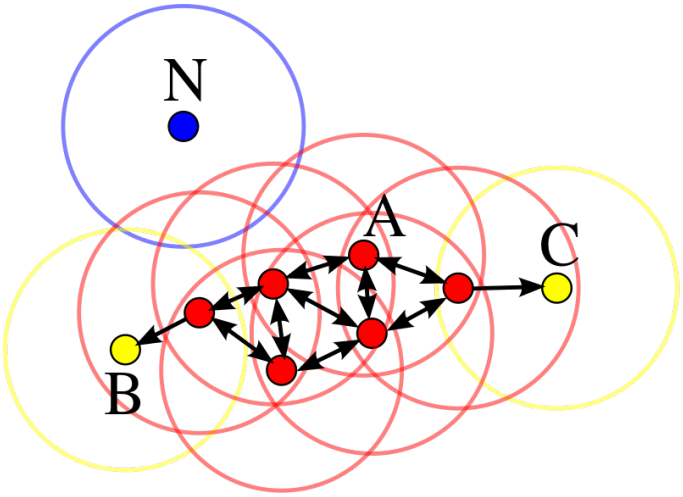


Figure 3.4: DBSCAN structures example, credits to `https://en.wikipedia.org/wiki/DBSCAN`.

points, since the area surrounding these points is an ϵ containing at least 4 points. Because they are all reachable from one another via a path, they form a single cluster. Points B and C are not core points, but are reachable from A via other core points. Hence, they belong to the cluster, and form the edge. Point N is a noise point that is neither core not directly reachable. With this knowledge, we can proceed to show the abstract algorithm of DBSCAN. It requires, as input parameters, ϵ and $minPts$. This algorithm is able to surpass k-means in some aspects. First and

Algorithm 3.4 DBSCAN

Find the points in the neighborhood of size ϵ of every point, and identify the core points with more than $minPts$ neighbors.
Find the connected components of core points on the neighbor graph, ignoring all non-core points.
Assign each non-core point to a nearby cluster if the cluster is an ϵ neighbor, otherwise assign it to noise.

foremost, as we introduced, it is able to find arbitrary shaped clusters, since we are not relying anymore on quadratic *formulae*. These were also the cause for the sensibility of k-means towards outliers, which DBSCAN is able to avoid thanks to the noise data-type.
The drawbacks for such an algorithm are related, from the ground-up, to the distance function used. As any clustering algorithm, it has inherit flaws related to the discovery of an appropriate ϵ value.

3.3 Dimensionality reduction

Along with clustering, the principal tool for unsupervised machine learning is dimensionality reduction. Also known as dimension reduction, it is the transformation of data from a high-dimensional space into a low-dimensional one. This transformation is able to retain some meaningful properties of the original data. It is useful in many different fields, and mainly thought for two reasons: to analyze data that are usually computationally intractable, due to size of complexity, and, in a similar way, to address the so called **curse of dimensionality**.
Coined by Richard E. Bellman[15], the curse of dimensionality can appear in many different fields, since it is related to an issue that various data have inherently. High-dimensional spaces, such as the one we are treating, are usually more complex to analyze with respect to three-dimensional physical spaces. Intuitively, when the dimensionality increase, the volume of the space increases so fast that the available data become sparse. To address this issue, a trivial solution is to add more observations, since that way data are more compact, but this cannot be considered a one-fits-all solution. In the machine learning field, some rule of thumbs have been created over the years[16], such as having at least five observations for each feature in the dataset. But these also are unreliable.
Going further from an intuitive approach, one of the main reasons for which curse of dimensionality is such an issue is related to the distance function used. A great summary for this issue is given by Pedro Domingos[17].

> "Our intuitions, which come from a three-dimensional world, often do not apply in high-dimensional ones. In high dimensions, most of the mass of a multivariate Gaussian distribution is not near the mean, but in an increasingly distant "shell" around it; and most of the volume of a high-dimensional orange is in the skin, not the pulp. [...] This is bad news for machine learning, where shapes of one type are often approximated by shapes of another."

Another development on this theme is a proof[18] which helps showing how much distance become insignificant in high dimensional scenarios. In particular, given a distribution on the real numbers $\mathbb{R}^d$, and any fixed n, it turns out that the difference between the minimum and the maximum distance between a random reference point q and a list of number n random data points $p_1, ..., p_n$ become indiscernible compared to the minimum distance. Given E an Euclidean distance function,

$$\lim_{d \to \infty} E(\frac{dist_{max}(d) - dist_{min}(d)}{dist_{min}(d)}) \to 0.$$

This is the usual proof that demonstrates why, in high dimensions, distance functions lose their usefulness.
Addressed why dimensionality reduction is so important in our scenario, we can proceed to describe it, and address the practical implementations we will try in the research.
Dimensionality reduction methods can be divided in two main categories:

- Feature selection methods. Such approaches try to find a subset of the input variables, and

can be done manually by filtering out non meaningful information, or automatically. It is useful to obtain a more accurate reduced space for regression or classification.

- Feature projection methods. These are techniques which *transform* data from the high-dimensional original space to one of fewer dimensions.

We will focus on the second family of approaches. The main concept behind all techniques is the one of creating a reduced set of features, known as feature vector. Among all, we will start from one of the most important technique, known as principal components analysis.

3.3.1 Principal component analysis

Principal component analysis[19] (PCA) is a dimensionality reduction technique, which allows to summarize a set of variables with a smaller number of representative variables, that are able to explain most of the variability in the original set. It is an unsupervised approach, meaning that it only involves a set of features $X_1, X_2, ...X_p$, and no associated response variable Y. Formally, the principal components can be seen as a collection of p vectors, where the i-th vector is the direction that best fits the data while being orthogonal to the first $i-1$ vectors. Other sources[2] rephrase this concept, underlying that principal component analysis seeks a small number of dimensions that are as *interesting* as possible. Interesting can then be defined by how much an observation varying along each dimension. We can see how each dimension is found, starting from the first principal component, following the approach from James, Witten, Hastie, Tibshirani[2] (2021, Chapter 12.1). Let $\phi_{11}, ..., \phi_{p1}$ be the so called *loadings* of the first principal component, which form the loading vector $\phi_1 = (\phi_{11}, \phi_{21}, ..., \phi_{p1})^T$. The first principal component of a set of variables $X_1, X_2, ..., X_p$ is the normalized linear combination of the variables with the first loading vector,

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + ... + \phi_{p1}X_p$$

The normalized attribute adds one more constraint, namely $\sum_{j=1}^{p} \phi_{j1}^2 = 1$. This is done to avoid the loadings getting large in a non useful manner. We are looking to maximize the observed quantity

$$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + ... + \phi_{p1}x_{ip}$$

which is the score of the first principal component. Loadings are found by maximizing the variance associated to the principal components, under the constraint.

$$\max_{\phi_{11},...,\phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1}x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

This problem is usually solved, in the realm of mathematics, via an *eigen decomposition*. The most common geometrical interpretation for the first principal component implies that ϕ_1 defines a direction in the feature space along which data vary the most. By then projecting $x_1, ..., x_n \in X$ in this direction, the projected value represent exactly the scores $z_{11}, ..., z_{n1}$, as for Figure 3.5.

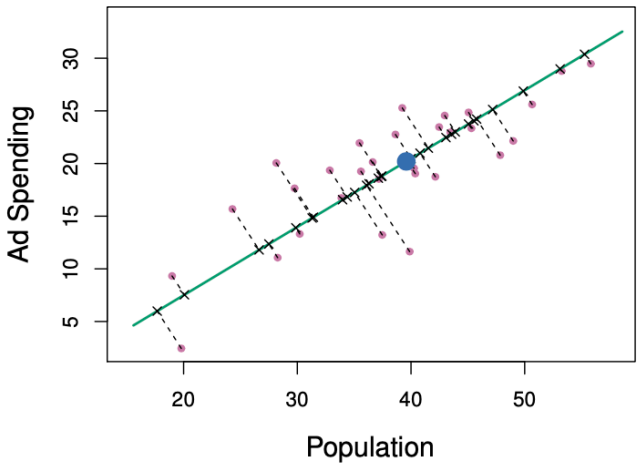


Figure 3.5: First principal component, plotted over a dataset about population and advertisement spending, from James, Witten, Hastie, Tibshirani[2] (2021, Chapter 12.1).

Then, the second principal component is the linear combination of $X_1, ..., X_p$ that has maximal variance out of all linear combinations which are uncorrelated with Z_1. Doing this, the scores $z_{12}, z_{22}, ..., z_{n2}$ are computed as

$$z_{i2} = \phi_{12}x_{i1} + \phi_{22}x_{i2} + ... + \phi_{p2}x_{ip},$$

where ϕ_2 is the second principal component loading vector. To find ϕ_2 we can solve the optimization problem from before, simply substituting ϕ_2 with ϕ_1. This constraints geometrically translate to the direction of ϕ_2 being orthogonal to the direction of ϕ_1. In a similar way, for larger datasets with $p > 2$, more distinct principal components can be computed following this same idea. Once all interesting components are obtained, they can be plotted to show a low-dimensional view of the data.

Computing all "interesting" components is not an easy matter, due to the fact that such a concept is by its nature personal and subjective. We know that by the definition of the loading vectors there are, for a dataset X of size $n \times p$, $min(n-1, p)$ distinct principal components. But, recalling our original aim, which is finding a way to capture the most information from a large set of features p with a subset of that information, we cannot consider them all. Typically, the choice of how many principal components obtaining is done via a *scree plot*. The way that the analysis works is looking for an elbow in the plot: once further principal components begin to explain less and less variance, then we can stop adding them.

4

Machine learning predictions

This chapter presents the results of the application of the techniques described in Chapter 3 to the analysis of NBA basketball data. The aim of this analysis is to find, in a dataset of NBA players described by their stat line in a year, an appropriate role, which should be more accurate than the notion of position. To this aim, we will focus on the clustering and dimensionality reduction, in the context of a classification problem.

4.1 Preparing the dataset

A first step required for the analysis is to prepare the data set. We will do so by filtering some information, going further with respect of what has been done in Chapter 2. The techniques we are going to use cannot be applied in presence of factor variables, meaning that we will need to create a subset of our original data set without the information of TEAM and POS. These are, as we recall, the team in which a player played during the referred year, and the position in which the player is recognized. Finally, we want to scale the numeric attributes, as this is a critical step. In particular, we apply the *scale* function provided by the *R* programming language. Given a data set, in our case a 2423×26, the root-mean-square of a column gets defined as $\sqrt{\sum(x^2)/(n-1)}$. For our problem, scaling data is fundamental due to the nature of our observations. For example, getting 10 rebounds is way more significant than getting 10 points inside a basketball game. A coach will prefer, almost always, a player with 10 rebounds with respect to one with 10 points. In a similar fashion, even greater is the impact obtained by a single assist: due to the three point revolution, in recent years an assist is worth roughly 3.47 points, factoring in the possibility of getting free throws for each shot attempt. Hence, we complete our pre-processing by scaling the data, meaning they are ready for the actual computation, starting with clustering. According to

different models, we will specify whether or not further pre-processing is needed.

4.2 Clustering methods

4.2.1 Centroid clustering analysis

As for centroid clustering, we will try to solve our problem of creating a categorization model via the use of the k-means algorithm. This decision comes from the fact that it is, indeed, one of the easiest ways to assess this problem. But, as discussed in Section 3.2.1, one of the biggest issues regarding centroid analysis, and hence k-means, resides in choosing the number of clusters. We obviously want a number which is not too high, otherwise our data in the resulting clusters would be too scattered, but, at the same time, we wish to get as much distinction as possible.
There are several ways to choose in a guided way this number, since there are a number of heuristics created for this sole reason. Three are particularly well known and used.

- Computing the within-cluster-sum of squared errors for different number of centers.
- Computing the silhouette value for different numbers of centers.
- Computing the gap statistic for different numbers of centers.

A data set with 26 variables which has not got through any reduction, due to the curse of dimensionality, is very likely to suffer the use of Euclidean distance, and this can be easily seen by looking at the plots for these methods. In each case it was applied the function *fviz_nbclust*, belonging to the *factoextra* package, external from the *R standard library*.

Listing 1 Call for the *fviz_nbclust* function.

```
fviz_nbclust(players.data.numeric.scaled, kmeans,
method = c("wss", "silhouette", "gap_stat"))
```

Figure 4.1 highlights the result obtained by within-cluster-sum of square errors. We can see, first and foremost, that there is not a clear elbow, meaning that in any case it will be particularly hard to find a meaningful point. The closest thing we can highlight is that, for two centers, the total within sum of square errors drops significantly, and after it goes down constantly. But this cannot be an answer to our problem. We need at least six categories to define players, otherwise we would end up with a similar distinction that the positions create. We hence deepen our analysis, looking at the silhouettes for different number of centers.

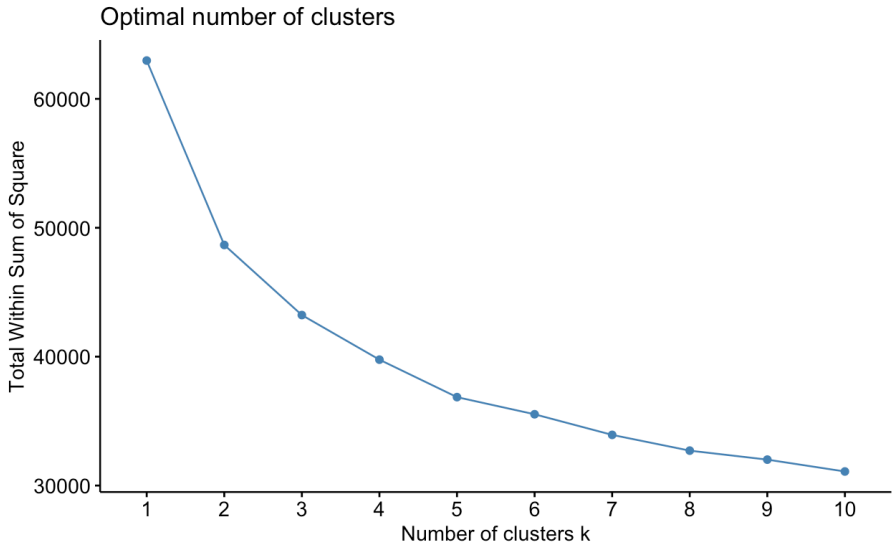


Figure 4.1: Results for the *fviz_nbclust* using the within-cluster-sum of squared errors.

Similarly to what we have seen for within-cluster-sum of squared errors, the silhouette analysis in Figure 4.2 suggests to choose a number of centers equal to two. We have already explained why this cannot be considered a meaningful choice, meaning we will conclude this first preliminary analysis looking at the results of the gap statistic.

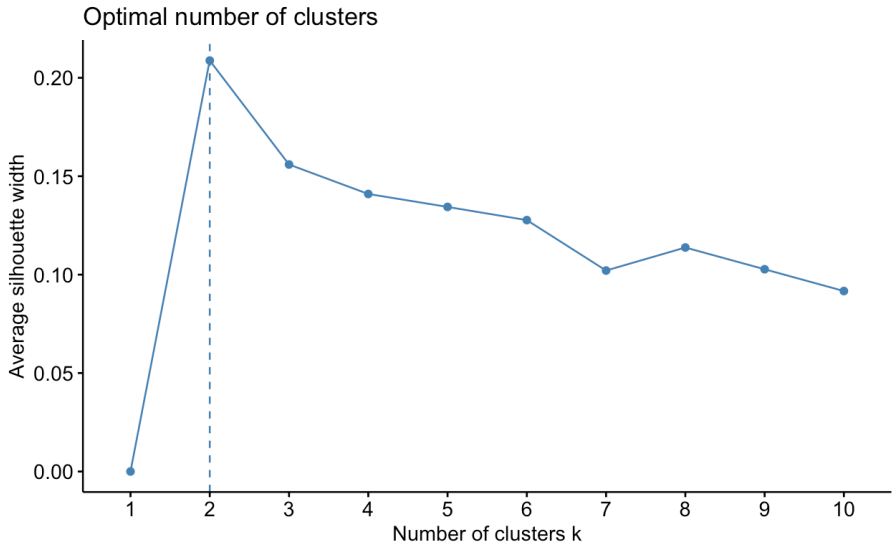


Figure 4.2: Results for the *fviz_nbclust* using the silhouette.

The interesting result of the analysis from Figure 4.3 deserves a bit of attention. Even with an high dimensionality, the gap statistic suggested to cluster our data in five different clusters. And while this is indeed not enough for our study, we want to underline that this is the exact number of positions currently used in the mainstream media when talking about basketball. Meaning our set of variables is for sure enough to explain all the positions classically used, but we need to push its limits in order to find a more clear distinction.

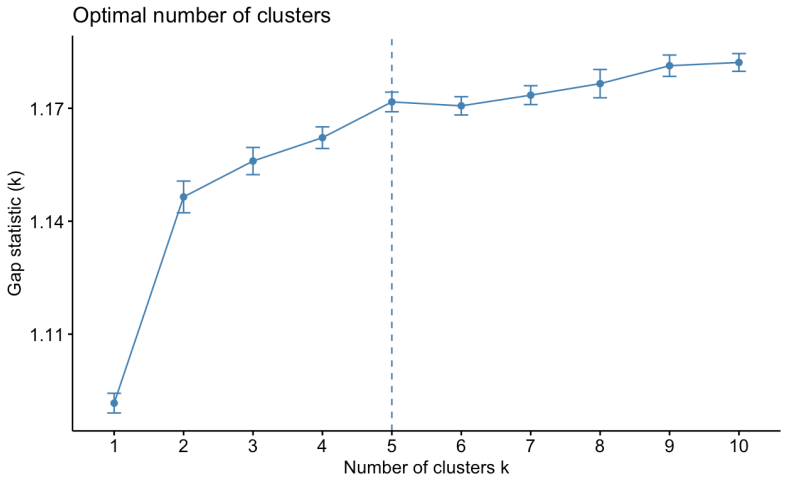


Figure 4.3: Results for the *fviz_nbclust* using the gap statistic.

For our purposes, we hence cannot choose any of the suggested number of centers, which puts the focus on how much of a complex matter is to choose the value k for clustering analysis.

OFFENSIVE ROLES \ DEFENSIVE ROLES	PERIMETER DEFENDER	AVERAGE PERIMETER DEFENDER	INTERIOR DEFENDER	AVERAGE INTERIOR DEFENDER	SWITCH DEFENDER	RIM PROTECTOR	D. REBOUNDER	NON-DEFENDER
PRIMARY BALL HANDLER	ELITE 2-WAY POINTGUARD	2-WAY ELITE POINTGUARD	ELITE 2-WAY POINT-BIG MAN	2-WAY ELITE POINT-BIG MAN	ELITE 2-WAY POINTFORWARD	ELITE 2-WAY POINT-CENTER	ELITE POINT-D. REBOUNDER	ELITE TRIPLE THREAT
PRIMARY SHOT CREATOR	ELITE 2-WAY POINTGUARD	2-WAY ELITE POINTGUARD	ELITE 2-WAY POINT-BIG MAN	2-WAY ELITE POINT-BIG MAN	ELITE 2-WAY POINTFORWARD	ELITE 2-WAY POINT-CENTER	ELITE POINT-D. REBOUNDER	ELITE TRIPLE THREAT
SECONDARY BALL HANDLER	2-WAY POINTGUARD	AVERAGE POINTGUARD	2-WAY POINT-BIG MAN	AVERAGE POINT-BIG MAN	2-WAY POINTFORWARD	2-WAY POINT-CENTER	POINT-D. REBOUNDER	TRIPLE THREAT
SECONDARY SHOT CREATOR	2-WAY POINTGUARD	AVERAGE POINTGUARD	2-WAY POINT-BIG MAN	AVERAGE POINT-BIG MAN	2-WAY POINTFORWARD	2-WAY POINT-CENTER	POINT-D. REBOUNDER	TRIPLE THREAT
SLASHER	2-WAY SLASHER	AVERAGE SLASHER	BIG SLASHER	AVERAGE BIG SLASHER	2-WAY SLASHER	2-WAY BIG SLASHER	2-WAY BIG SLASHER	SLASHER
PASS-FIRST PLAYER	ELITE 2-WAY PLAYMAKER	2-WAY ELITE PLAYMAKER	ELITE 2-WAY BIG-PLAYMAKER	2-WAY ELITE BIG-PLAYMAKER	ELITE 2-WAY PLAYMAKER	2-WAY ELITE BIG-PLAYMAKER	ELITE POINT-D. REBOUNDER	PASS-FIRST PLAYER
SCORER	ELITE 2-WAY SCORING GUARD	2-WAY SCORING GUARD	ELITE 2-WAY SCORING BIG MAN	2-WAY SCORING BIG MAN	ELITE 2-WAY FORWARD	ELITE 2-WAY SCORING CENTER	SCORING AND REBOUNDING	DOUBLE THREAT
OFF-SCREEN SHOOTER	ELITE 3&D	3&D	ELITE BIG 3&D	BIG 3&D	ELITE 3&D	ELITE 3&RIM PROTECTOR	STRETCH BIG MAN	SHARP SHOOTER
SPOT-UP SHOOTER	ELITE 3&D	3&D	ELITE BIG 3&D	BIG 3&D	ELITE 3&D	ELITE 3&RIM PROTECTOR	STRETCH BIG MAN	SHARP SHOOTER
PURE SHOOTER	ELITE 3&D	3&D	ELITE BIG 3&D	BIG 3&D	ELITE 3&D	ELITE 3&RIM PROTECTOR	STRETCH BIG MAN	SHARP SHOOTER
POST-UP CREATOR	2-WAY POST-UP POINTGUARD	POST-UP POINTGUARD	ELITE 2-WAY POINT-BIG MAN	2-WAY ELITE POINT-BIG MAN	ELITE 2-WAY POINTFORWARD	ELITE 2-WAY POINT-CENTER	ELITE POINT-D. REBOUNDER	POST-UP CREATOR
POST-UP SCORER	2-WAY POST-UP GUARD	POST-UP GUARD	2-WAY POST-UP BIG MAN	POST-UP BIG MAN	2-WAY POST-UP FORWARD	ELITE RIM PROTECTOR	BIG MAN	POST-UP SCORER
PICK AND ROLLER	2-WAY SCREENER GUARD	SCREENER GUARD	ELITE RIM RUNNER	RIM RUNNER	2-WAY FORWARD	ELITE RIM PROTECTOR	RIM RUNNER	ROLLER PLAYER
ROLL&POP	2-WAY BIG GUARD	BIG GUARD	ELITE 2-WAY DOUBLE THREAT BIG MAN	2-WAY DOUBLE THREAT BIG MAN	ELITE 2-WAY FORWARD	ELITE 2-WAY DOUBLE THREAT BIG MAN	ELITE BIG MAN	ROLL&POP PLAYER
POST&3	2-WAY POST-UP GUARD	POST-UP GUARD	ELITE 2-WAY DOUBLE THREAT BIG MAN	2-WAY DOUBLE THREAT BIG MAN	ELITE 2-WAY FORWARD	ELITE 2-WAY DOUBLE THREAT BIG MAN	ELITE DOUBLE THREAT BIG MAN	POST&3 PLAYER
OFF-BALL PLAYER	ELITE PERIMETER DEFENDER	PERIMETER DEFENDER	ELITE INTERIOR DEFENDER	INTERIOR DEFENDER	ELITE ALL-AROUND DEFENDER	ELITE RIM PROTECTOR	D. REBOUNDER	OFF-BALL PLAYER
MARGINAL OFFENSIVE ROLE	PERIMETER DEFENDER	AVERAGE PERIMETER DEFENDER	INTERIOR DEFENDER	AVERAGE INTERIOR DEFENDER	SWITCH DEFENDER	RIM PROTECTOR	D. REBOUNDER	MARGINAL PLAYER

Figure 4.4: Classification of advanced roles offensively and defensively, provided by Hack a stat.

The website Hack a stat created the helpful guide from Figure 4.4 to define advanced roles in

basketball. We want to focus on the left column, where the offensive roles are defined, since, for the scope of this research, we won't consider defensive parameters. We are able to see, distinguished by different colors, eight different macro categories, which are then further divided. But, as a matter of fact, we want to focus solely on the macro distinctions, since the smaller ones are hard to grasp given the nature of our data set. As explained in Chapter 2, we do not have access to advanced offensive metrics, which implies we cannot distinguish, for example, between a post-up shooter and a pure shooter. Not having access to the Post-up% for each player would imply that while a distinction between these players **does** exist, they could only be classifiable as "shooters". The only further assumption cluster we will assume, other than the eight suggested by the source, is the superstar cluster. We already talked about how much in NBA the presence of star players is fundamental in order to create a successful team, and found out in the preliminary analysis that there is indeed the presence of more versatile player than others, thanks to the *VI*,versatility index, information we have in our data set. With all of this being said, we proceed to analyze the k-means clustering with nine centers, done without dimensionality reduction, with the three algorithms that the *R* programming language offers, starting from its default option.

Hartigan-Wong

For purposes of reproducibility for this experiment, we will always set the seed for the analysis at 123. That being said, we can proceed to the first call to the *kmeans* function.

Listing 2 Call for the *kmeans* function using the Hartigan-Wong heuristic.

```
km.res <- kmeans(players.data.numeric.scaled,
centers = 9, iter.max = 50, nstart = 25,
algorithm = "Hartigan-Wong", trace = FALSE)
```

Listing 2 can be explained as follows: we are looking at our dataset, filtered and scaled, trying to find on it nine clusters. The number of iterations is usually set to 10, but due to the high dimensionality of this dataset it had to be increased.
After the execution, we end up with a clustered version of the dataset, which cannot be visualized on its own, since 26 dimensions cannot be plotted. To solve this issue, we resolve to dimensionality reduction, thanks to the function *fviz_cluster*, belonging to the *factoextra* package. We hence plot the nine found clusters on two dimensions, to analyze the result obtained.

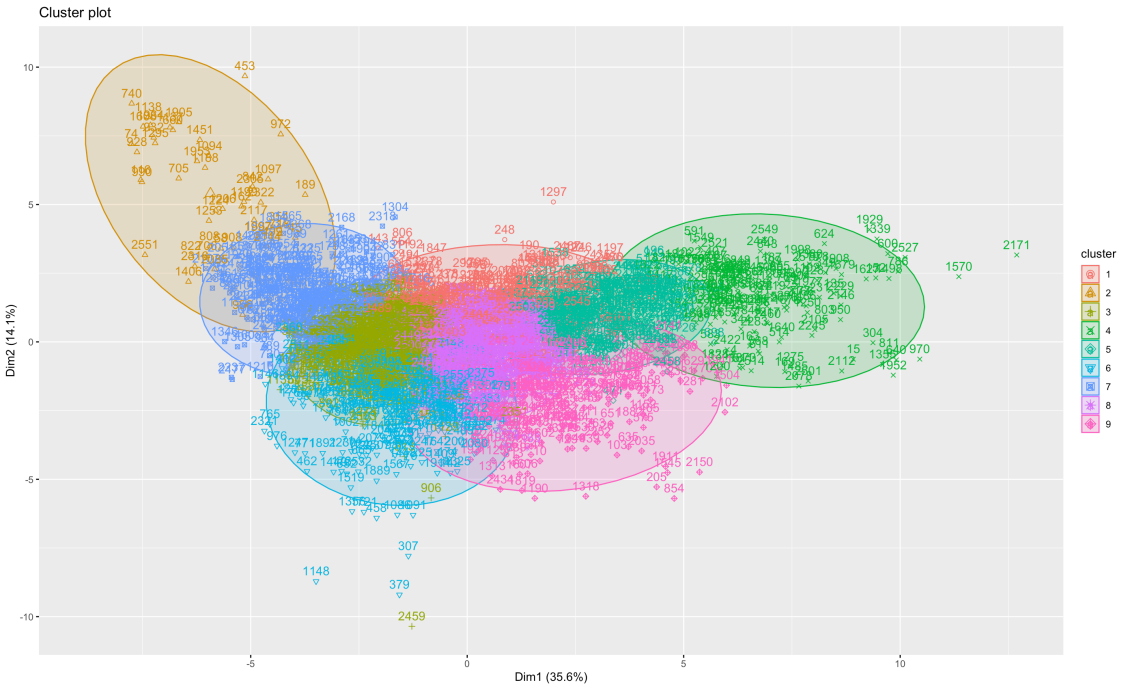


Figure 4.5: Hartigan-Wong heuristic result

A first way to address the results from Figure 4.5 is to look graphically at the clusters. We see that there are roughly four of them which are precise, being 2, 4, 5 and 9. These have their observations rather independent, while the remaining ones mix with one another, which hints to a degree of between-cluster-dissimilarity. In other words, observations may be very similar to one another in these groups. We can look closer at the within sum of squares with Table 4.1.

Table 4.1: The Within-Cluster-Sum of Squared Errors for Hartigan-Wong heuristic

Cluster number	WSS
1	3334.156
2	1481.673
3	5167.883
4	2590.099
5	2611.655
6	4773.328
7	4702.025
8	4044.474
9	3117.240

Table 4.1 shows that clusters 2, 4, 5 and 9 are the ones with the more contained within sum of squares, which implies that players inside those groups are more similar to one another and less similar to the ones outside them. Hence, this result confirms our initial hypothesis obtained by looking at the plot.
Before addressing a comprehensive analysis of the results, we would like to explain the methodology used for this purpose. For clusters which are hard to analyze, a subset was chosen, based on the mean of games and minutes played in the league, as for official NBA statistics. This implies players were discarded when they played less than 20 minutes per game and 40 games per season. For clusters 3, 6 and 7, these requirements were too much, and the results were almost empty classes. This alone is an index of the fact that in here we will find mostly role players, which do not find much times in games and hence will need a thorough analysis. In cases like cluster 8, instead, this was a very useful mean of understanding characteristics of the players.
To get a complete picture and understand how we can get better results, we proceed now to analyze each one in detail.

- Cluster 1: composed almost entirely by guards, this cluster contains what we called the ground generals. In particular, they are smaller-shaped players who are aimed at developing plays for their teammates, generally by providing assists. This is indicated by a high AST and AST%. A good example for this cluster is Draymond Green, which has the shape of a Forward but is known for his assists proficiency on the court, meaning that not only trivial players are being included. Another interesting result is the presence of a center, such as Marc Gasol, which can be seen as the "father" to the modern center role, without ever fully achieving it.
- Cluster 2: this cluster is particular, since it contains relegated players. This is easily noticeable by the fact that, overall, they played 8 minutes per game, probably garbage time. With this definition we refer to minutes played in games where the score is almost decided, and hence coaches resort to use players who usually stand in second lines. Among all of the listed players, only two of them are still in the NBA, meaning that finishing in this cluster is a good way of detecting wheter a player needs to be cut off a team.
- Cluster 3: we see a high presence of players who still need development and more time in the league. Predominantly, we find shooters, and they are less likely to be close to the relegation cluster. This can also be highlighted geographically, since the cluster is in between cluster 2, and the more meaningful ones. Yet again, an analysis of this cluster can show that it is filled by roughly 80% with guards and forwards. A proper analysis of the forwards present in this cluster show that they are usually shooters, either from 2 or 3 points, meaning that they are less physical players, but preferably shooters.
- Cluster 4: this can be summed up as the superstardom cluster. It is one of the easiest to analyze: we find in here the most valuable players of each season, as well as the best, well-rounded players in the league. They are most of the times players good at pretty much anything, meaning that a rough classifier will almost any times classify them together. The VI plays for sure a major role in this distinction, as we have seen in Chapter 2. We would like to address players we described as "shooting centers" are all present in this cluster. For this reason, one of the most coveted weapons in modern basketball is classified almost always with a superstar label. The reason is simple: a shooting center will be good at high percent 2 point shots, will be close to the league mean in 3P%, all while take rebounds in a game. This will, by default, boost their VI.

- Cluster 5: contains the role players regarded as good 3 point shooters in the league. This can be seen by the fact that their mean on 3P% is roughly 37%, higher than the mean of the elite shooters in the league. This aligns with the initial analysis showing that this cluster was more independent with respect to the others.
- Cluster 6: big framed players with less playtime, still very proficient in rebounding. Again as in cluster 1, we can see that this classification is working even for not "classic" rebounders, which are centers and big forwards, by looking at the guards which are present in the cluster. We see in here more physical guards, such as Patrick McCaw, Gary Payton II and John Konchar, which are all known for their athleticism.
- Cluster 7: similarly to cluster 3, we see a high presence of players who still need development and more time in the league. Predominantly, we find drivers, and players who are more likely to become belonging to cluster 8 with more development and minutes in their favor. Again, this cluster differ from number 3 with respect to the preferences that the players have in generating scoring. While cluster 3 players prefer shots and assists, these ones will rely more on free throws and rebounds.
- Cluster 8: cluster containing a wide variety of players, and indeed one of the most populated ones, mostly of which are proficient inside the paint. Hence they are most of the times on-ball players who prefer to try drives to the rim, or two point shots. This can be seen by the percentage on 2P%, close to the average of the league, and FT%. The attempts for these two types of attacking shots need to be adapted: these players, since they average less time on the court, cannot expect to attempt the same number of free throws or 2 point shots as the starting five of each team, since they will also be less likely to be the focal point of attack for each team.
- Cluster 9: finally, we find the classic center cluster. This is easy to see due to the fact that, for example, there are no guards and very few forwards in this group. The only exception comes from Brandon Clarke, listed as a Forward Guard, but this can be seen as a mistake on the dataset, since it is a clear example of big framed, close to the rim, forward. We do not find only trivial centers, but also guards with a preference towards rebounding and high-percentage 2 point shots. Indeed, the mean for 2P% is 60% in this cluster, meaning we find players with highly efficient shots, which imply they stay closer to the rim.

A final result we want to address is how much clusters are balanced between one another with respect to number of members, to understand possible trends and preferences in the modern NBA. Indeed, we can gather very interesting results from Figure 4.6 alone. Firstly, that the most crowded cluster is number 3, containing role players for 2 and 3 point shooting. This is yet another proof to the concept we tried to explain in Chapter 1: the three point revolution is the most present trend of the last years in NBA basketball, and teams are trying their hardest to achieve excellence, by gathering a large number of shooters in order to find efficiency in this department. In a similar fashion, players from cluster 8, which is the second most crowded, are fundamental. Drivers to the rim, and generally multi-layered second unit players, who can do almost anything, that enter the game when the superstars from cluster 4 are tired. The latter is indeed the least crowded among the clusters which meaningful players for the obvious reason that superstars are a rare exception, and not the norm in the NBA. Other interesting trends can be seen in the fact that ground generals and classic centers are less populated, meaning that we are moving from static roles with two or three precise specialties to more fluid ones. Nowadays each player is supposed

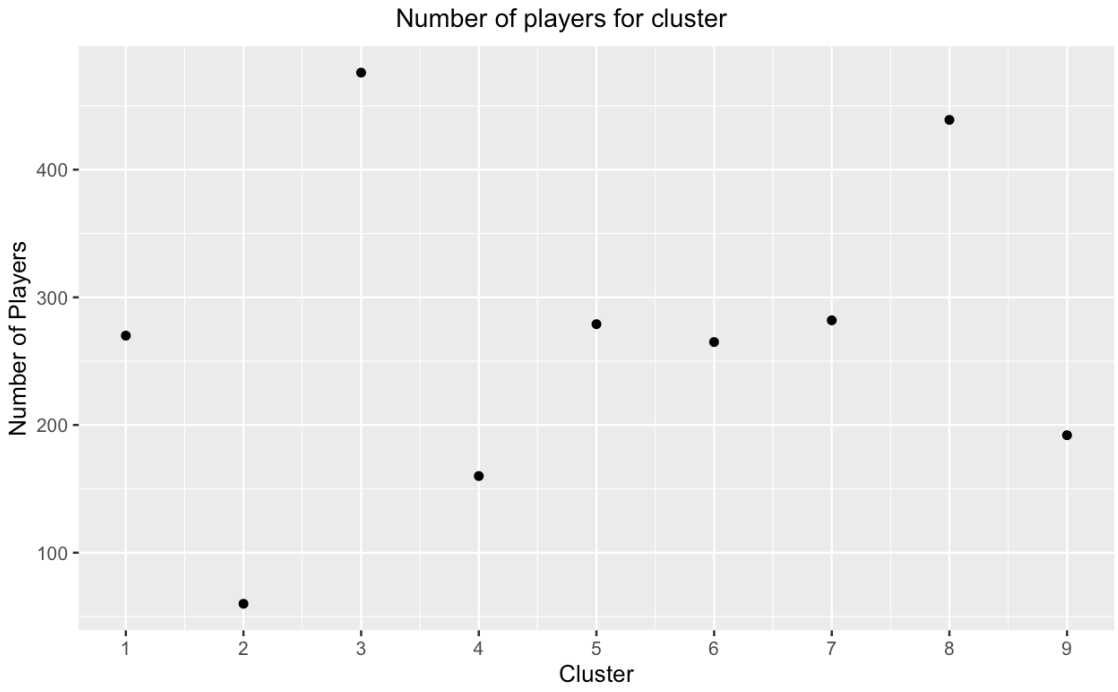


Figure 4.6: Balance of clusters size.

to do almost anything at a good level. Still, they are fundamental in today's NBA: any team still want a specialized game creator which dishes assists, and a big man who can gather rebounds and score in an efficient manner.

Lloyd-Forgy

This heuristic of k-means will follow all the previous assumptions made for the Hartigan-Wong analysis.

Listing 3 Call for the *kmeans* function using the Hartigan-Wong heuristic.

```
km.res <- kmeans(players.data.numeric.scaled,
centers = 9, iter.max = 50, nstart = 25,
algorithm = "Lloyd", trace = FALSE)
```

The call to the function *kmeans* from the *R library* remains the same, with the exception of the algorithm used, which of course becomes "Lloyd". All the other parameters remain the same, since the dataset did not undergo any other change, and the number of centers is the same for the reasons explained before. All of this said, we proceed to show the output for the algorithm, plotted again in two dimensions.

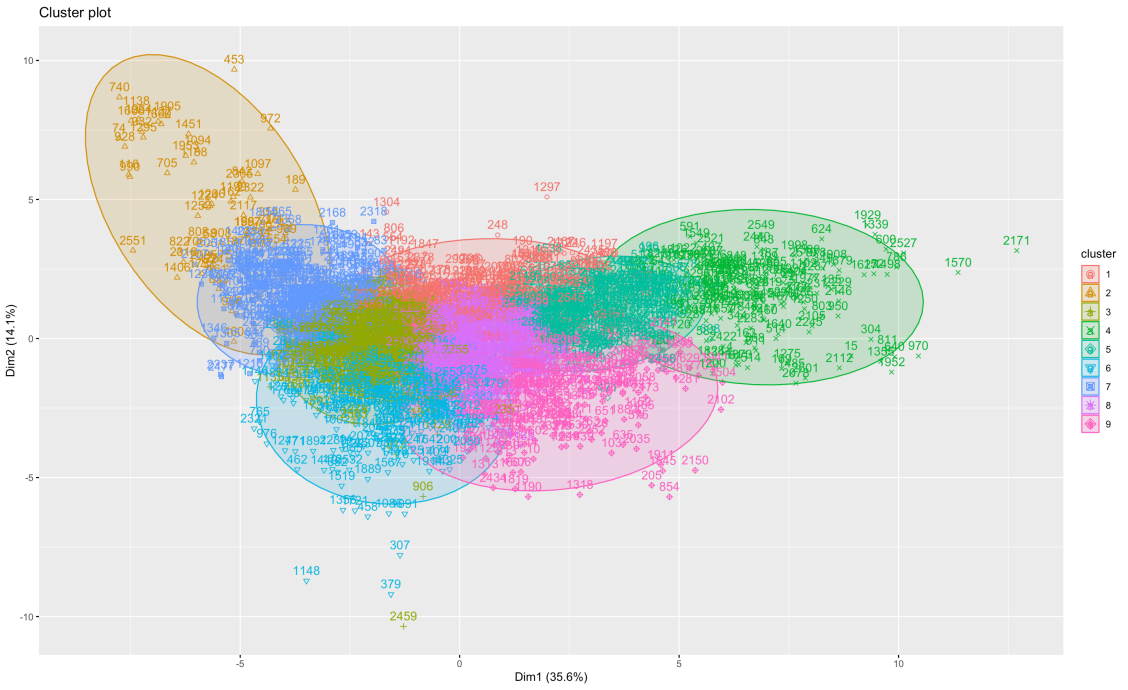


Figure 4.7: Lloyd-Forgy heuristic result.

The results from Figure 4.7 prove to be especially similar to the Hartigan-Wong heuristic. This comes from the two being particularly similar, meaning overall we cannot expect to have much more different results. We can see graphically how much the two methods differ from one another thanks to a free online tool, Diffchecker.
Graphically, as shown in Figure 4.8, we cannot infer any particular difference in the classificators: the generated ellipses look to be slightly larger in the Hartigan-Wong method, but apart from that even the outliers look to be the same. About this topic, we want to spend a few words talking about them, to understand if the classificator is failing, or where the dataset may still be incomplete or wrong.

- 1297: Derrick Walton Jr, 2021.2022, playing for the Detroit Pistons. While he can be indeed be listed as a floor general, averaging 7 APG with a 27% AST% for his team, what makes him a sort of outlier for his cluster was his abysmal 2 point shooting, one of the worst of the league. This made him end up further from the clusters with good shooters, while still belonging to the ground generals.
- 2459: Jordan Sibert, 2018-2019, playing for the Atlanta Hawks. Listed as belonging to cluster 3, played only for 4 minutes during the whole season, but his ORTG was computed nevertheless, despite the evident lack of substantial information. It can be safely removed from the dataset.
- 1570, 2171: James Harden, 2019-2020, playing for the Houston Rockets. What sets this observation as an outlier is the immeasurable metric Harden obtained during this particular season. Indeed, Harden shot 800 free throws during the regular season, an all time record,

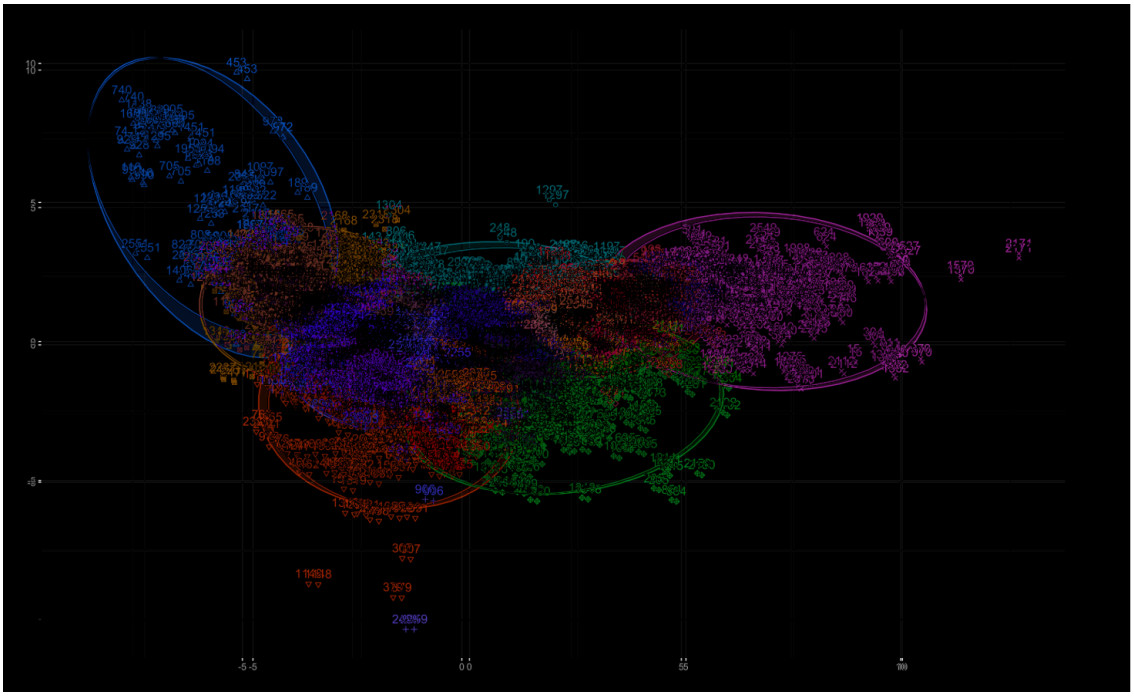


Figure 4.8: Difference checking results between Hartigan-Wong and Lloyd-Forgy heuristics, via Diffchecker.

with an outstanding 87% efficiency. The reason for this point being an outlier is, hence, due to this player pushing a common statistic to its breaking point. James Harden, 2018-2019, playing for the Houston Rockets. The same exact line of reason as before applies, with Harden shooting 857 free throws, with an 88% efficiency. In this case, he ended up winning the trophy for best player in the season, since even in other metrics his offensive efficiency was so outstanding to set many records for years to come.

- 307, 379, 1148: Damian Jones, 2020-2021, playing for the Los Angeles Lakers. He was shooting with a very interesting percentage of success, roughly 94% with 2 shots a game, but they were not enough to categorize him as a shooter, and ended up for his frame and rebounds to be listed as an athletic center, belonging to cluster 6. Patrick McCaw, 2020-2021, playing for the Toronto Raptors. He played roughly 30 minutes the whole season, with 5 minutes per game, and ended up shooting just one shot, which went in, keeping him at a 100% efficiency. Obviously his presence is debatable in this dataset, but for the scope of our work we do not consider him interesting enough as a sample, and we resort to remove him. Reggie Perry, 2020-2021, playing for the Indiana Pacers. The contributions he provided to the team were obviously not enough, since, after his move to the Pacers he ended up playing only 9 minutes, and his ORTG was computed, and is one of the highest in the dataset, at 200. As for the other samples, we resort to remove him.

We can hence conclude that the outliers are due to two main factors in our classificator. The first being the players averaging statistics which are outstanding, or polarized to one specific strenght of a player's game, as we have seen for Harden and Walton Jr respectively. The second

reason is due to the dataset, which list an ORTG for players whose contributions to their team were limited at best, resorting in extreme situations as we have seen. Recall that, in Chapter 2, we excluded from the dataset any player for which ORTG was not computed. With this analysis, we conclude that our classificator is not having a clear case of errors in the outliers, and proceed to analyze the within sum of squares, comparing the results for the Hartigan-Wong and Lloyd method.

Table 4.2: Comparison for Within-Cluster-Sum of Squared Errors between the nine clusters .

	1	2	3	4
Hartigan-Wong WSS	3334.156	1481.673	5167.883	2590.099
Lloyd-Forgy WSS	3408.436	2025.303	4975.658	2683.734
5	6	7	8	9
2611.655	4773.328	4702.025	4044.474	3117.240
2596.632	4793.108	4415.512	3969.081	2969.687

We can see that the within sum of square remains pretty much the same, with slight variations considering the overall values. The Lloyd method is less precise in cluster 2, which wasn't an issue to begin with, since it was the easiest to classify.

MacQueen

We are looking at the final proposed implementation for the standard library of k-means by the *R language*. Being MacQueen solution different in a measure from the previous ones, the parameters for the call need some tuning.

Listing 4 Call for the *kmeans* function using the MacQueen heuristic.

```
km.res <- kmeans(players.data.numeric.scaled,
centers = 9, iter.max = 100, nstart = 25,
algorithm = "MacQueen", trace = FALSE)
```

The call to the function *kmeans* from the *R library* here has seen mainly the change in number of iterations needed to reach convergence. The previous number of 50 was not enough anymore, and we decided to double it, to be completely sure of not missing any meaningful information during the call. Apart from this, the heuristic used has changed, hence we show now the results for the aforementioned call.
Figure 4.9 shows that we are not able to see pretty much any difference with the previous results. While graphically we are in presence of an almost identical result to the two before, we want to dig even deeper in this case, and find out whether or not there are any significant changes in the results. Firstly, we do so by taking a look at a complete version of the WSS table.

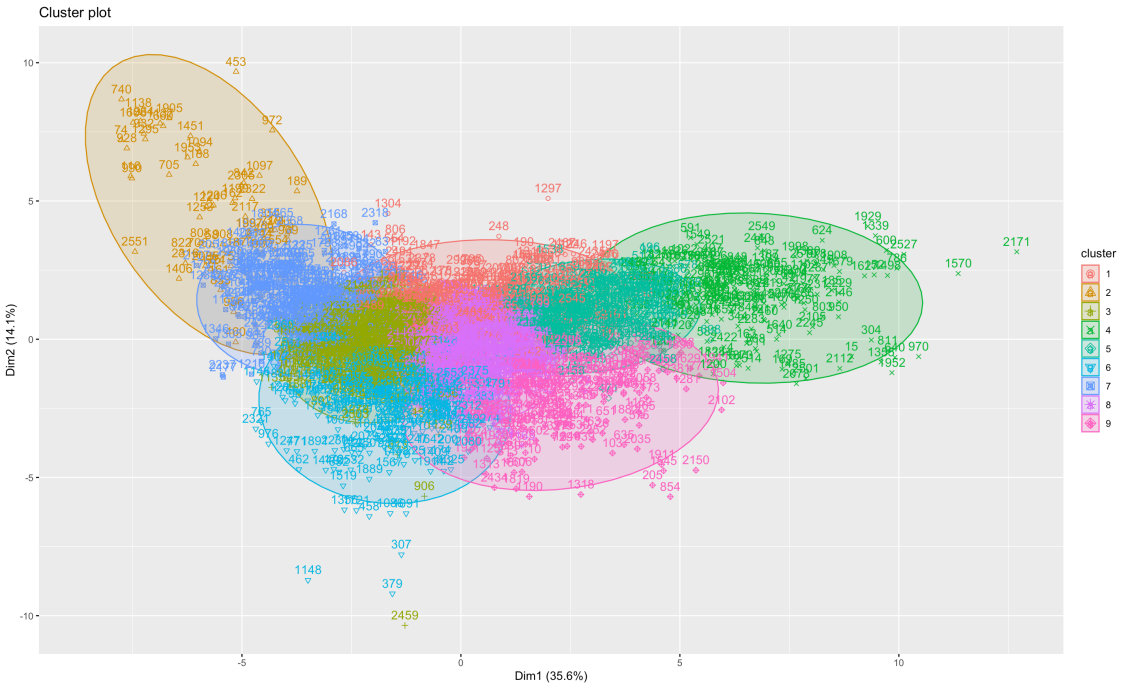


Figure 4.9: MacQueen heuristic result

Table 4.3 indeed shows that the decision for the heuristic used is not as clear as one could expect: all the proposed solutions tend to line up on defined results, and apart from the occasional errors and diversions on, for example, cluster 2, we are not able to say that one is clearly better than the other. Being it the default choice for the *R* programming language, we will from now on refer to Hartigan-Wong result as the main one.

Table 4.3: Comparison for Within-Cluster-Sum of Squared Errors between the nine clusters considering all heuristics.

	1	2	3	4
Hartigan-Wong WSS	3334.156	1481.673	5167.883	2590.099
Lloyd-Forgy WSS	3408.436	2025.303	4975.658	2683.734
MacQueen WSS	3394.839	1979.975	4970.335	2855.353

5	6	7	8	9
2611.655	4773.328	4702.025	4044.474	3117.240
2596.632	4793.108	4415.512	3969.081	2969.687
2695.762	4799.396	4233.201	4016.438	2900.387

Finally, we wanted to explore if this result and the main one, from Hartigan-Wong, diverge conceptually. The *dplyr* library allows to examine the different values referring to them as sets, and we are able to use it in order to understand where the algorithms diverge. This analysis has been

carried on by looking at the "misplaced" players: the observations which were categorized in a cluster from Hartigan-Wong's method, and then were differently treated by MacQueen's one.
The conclusion is that the algorithms do not diverge conceptually, and all the found clusters are identical to the ones described in Hartigan-Wong's solution. The only instances of different categorizations were the ones which were actually more thin-lined. Players such as Kyle Anderson, who can indeed be classified as a center, as well as a driving forward on-ball, or Evan Fournier, which is categorized as a guard, and is very proficient both in shooting and assisting his teammates.
All the discussions above prove that, at least in this realm, changing the k-means algorithm is unlikely to have any meaningful impact on the results of our experiments. As a positive consequence, we are also able to underline that k-means, for our purposes, is stable in each of its implementations, and will, almost anytime, provide safe results. In some cases it can even be useful to execute different algorithms, to get a more clear view for players which are hard to be defined in just one category, like the ones listed above.

Analysis of the superstardom cluster

During the analysis done in section 4.2.1, we were able to discover a particular set of players which was briefly discussed, and requires, as we believe, closer attention. The superstar cluster indeed contains a wide variety of players, and we can safely assume that they were grouped due to their VI. This variable explains if a player can be considered a high-level one, and it measures the versatility of their playstile. Indeed, superstar players excel at almost any aspect of the game, but they too have their specialties: Stephen Curry, who is famous for his 3 point shot, is in the same category as Giannis Antetokoumpo, which main capability is his presence under the rim. Hence, what we aim to achieve in this section is to find a second level of clustering, by applying some of the techniques analyzed so far to the cluster number four obtained in section 4.2.1.
We will adapt on this smaller scale all the preparations that were done in the previous analysis. In particular we will use a slimmer dataset, where we remove the following variables: X, AGE, TO%, eFG%, TS%, SPG, BPG, TOPG, ORTG, DRTG, MIN%, USG%, cluster. We will then also be sure to scale the variables, as it is a step required by all the proposed methods for the analysis. We will focus solely on the *kmeans* approach, utilizing the Hartigan-Wong algorithm. First of all, we have to understand the optimal number of clusters for this new subset of a dataset, and we will do so by analyzing it with the gap statistic method. We have seen before in section 4.2.1 that this analysis provided the closest results to the value we used in the end. And while it is also the most expensive in computational terms, we can safely use it, due to the very reduced number of players that we need to analyze. The results from this first research can be seen in Figure 4.10.
The result is straightforward in its analysis, meaning we will choose as our value k, number of clusters, three. As for the number of iterations, 50 can be considered almost too high for the means of this experiment, but it also allow us to avoid missing any information. The heuristic, as we concluded before, does not meaningfully influence the final results of the algorithm itself. In Figure 4.11 we find the results for this experiment, where the seed was again set to 123.
Again, we find the outcome particularly interesting, since it allows us, first of all, to have a deep

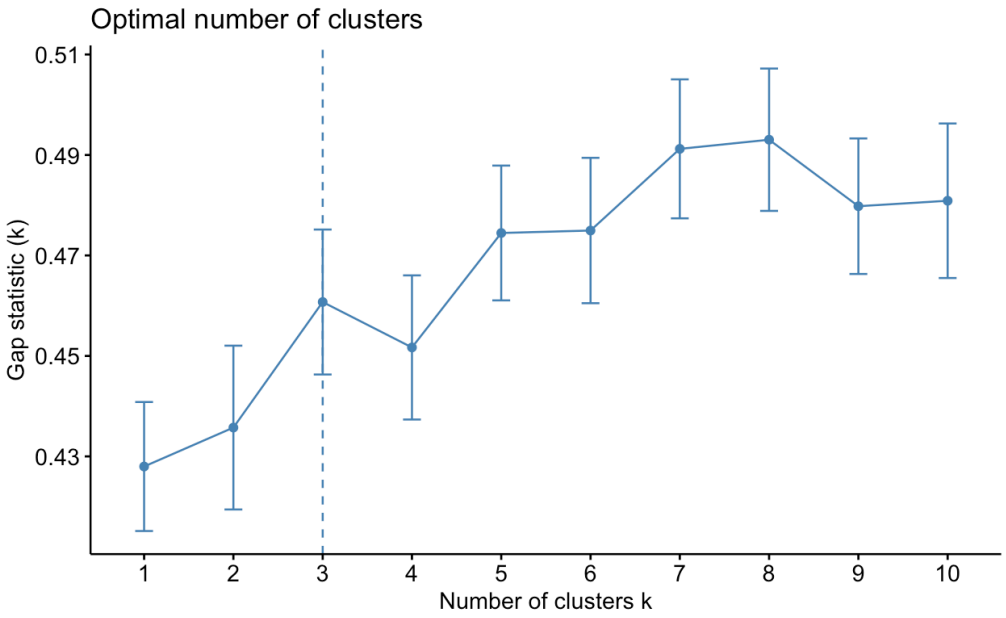


Figure 4.10: Results for the *fviz_nbclust* function using the gap statistic, applied to cluster 4 found in section 4.2.1

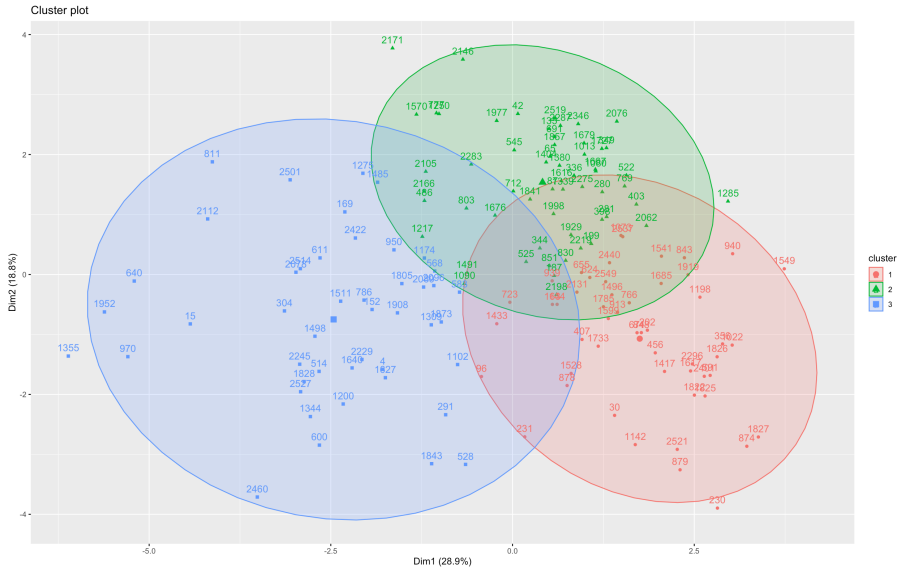


Figure 4.11: Results from applying k-means on the cluster 4 obtained in section 4.2.1.

look inside a cluster. Inside of it, there are indeed some sorts of natural clusters, which are highlighted by the ones in the results. Indeed, even if the clusters overlap with one another, this is due to the nature of k-means, which can only generate elliptical shapes. Before going in details of

Table 4.4: Within-Cluster-Sum of Squared Errors for the clusters found by applying k-means to cluster 4 obtained in section 4.2.1.

	1	2	3
WSS	490.23	386.21	591.32

the found clusters, we can take a look at the within-cluster-sum of squared errors for the resulting clusters.
The results, again, prove to be extremely encouraging, and are by far lower than the ones we found during the previous experiments. With this consciousness, we can finally analyze the results cluster by cluster. Before proceeding, we want to notice that the clusters are balanced as well in their sizes, going from 46 observations to 58.

- Cluster 1: this category of superstars includes the ones which are not seen as the main scoring terminal of their teams. While there are exceptional players in terms of efficiency from 2 and 3 point shooting, such as Jimmy Butler in the first case, or Kyrie Irving in the second, they are not the players responsible to carry their team's offensive maneuvers. They are mainly guards, due to the fact that they are expected to act as selfless players, able to dish assists to their teammates, as the presence of Derrick Rose during the 2019-2020 season can show.
- Cluster 2: in here we are able to find some of the most important players in any team's offensive efforts. These are the most extremely efficient 2 or 3 point shooters, players such as Stephen Curry, Devin Booker or James Harden. We are able to see miscellaneous players such as guards or forwards inside this cluster. This serves to prove even further that the main mean of offense in recent years is, indeed, the three point shot.
- Cluster 3: while the three point shot has seen an increasing attention during the last years due to the development in its productivity, we still have to recall that the most efficient shots in basketball are those taken closer to the rim. In this final cluster we find such players, who play significantly closer to the rim, usually shooting, and are as well able to gather a large amount of free throws by driving and dribbling. Also, there are the players with the highest efficiency ratio, due to their high usage and the type of shot they take. In here, there are mainly centers and forwards. Some examples are players such as Nikola Jokic, Julious Randle or Lebron James.

Another further note on this discussion is on how the same player, from season to season, is not always categorized in the same cluster. For example, Derrick Rose was a fundamental asset in 2019-2020 for the Detroit Pistons, and the sole player who could carry assists to the rest of his teammates. Instead, on following years, he was categorized as primarily a floor general. This is an interesting consideration, which could incentives the use of this tool also during parts of a season, in order to understand the development of a player.
We conclude this section by stressing that what we have seen could indeed be a further method of analysis for all the found clusters in previous sections. It could help reducing the margin of error, as we have seen in Table 4.4. There is, in this method, the inherit risk of creating too many meaningless categories. We decided to show this analysis for the particular case of the superstar cluster, being it in the gray line of being a meaningful, but not detailed enough result.

4.2.2 DBSCAN analysis

To go beyond the main issues of centroid clustering, we defined in Chapter 3 the concept of density-based clustering, introducing the DBSCAN routine. This allows to implement a nonparametric function, in which we do not have to specify the number of clusters we want to obtain. Also, it will help us understanding whether or not there are natural clusters inside the original dataset we are using for this study.

While not utilizing a parametric approach, the function still needs two important input values, which, as we explained in Chapter 3, serve the purpose of understanding which are the core points of our datasets. These are $MinPts$ and ϵ. These are usually found in the order we propose here. In particular, $MinPts$ follows a rule of thumb, which is pretty simple, and goes by $MinPts \geq D+1$, where D is the number of dimensions in our dataset. Given that there are in total 26 numerical variables, we will assume for a first example a value of $MinPts = 27$. Another usual configuration is $MinPts = 2 \cdot D$. On the other hand, ϵ is a bit less straightforward to obtain. We want to find, given each point and its k-nearest neighbors, a value of the mean distance which is high, but not too big to handle. To do this, these k-distances are plotted in an ascending order, and the aim is to determine the "knee", which corresponds to the optimal ϵ parameter. A knee corresponds to a threshold where a sharp change occurs along the k-distance curve. Inside the *dbscan* package, we utilize the *kNNdistplot* function, which is used in this case to draw the k-distances plot. In this experiment, $k = MinPts = 27$.

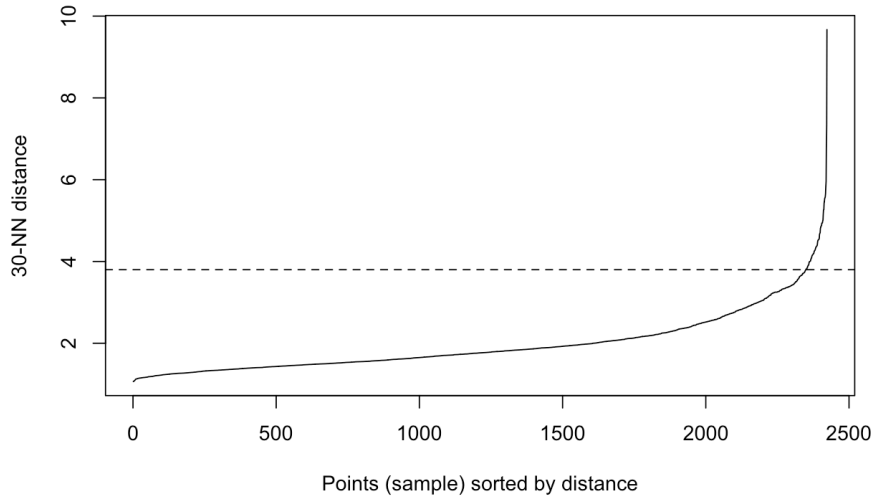


Figure 4.12: K-nearest neighbors distances plotted in ascending order.

The elbow, as Figure 4.12 shows, is at, roughly, 5.5, and is indicated by the horizontal dotted line. For the experiment, this will be the value of ϵ. These distance may seem low considering our dataset, but we recall that, for all the analysis in this chapter, we are using the scaled version of

our dataset, in order to avoid problems related to variability of the observations.
In the *R programming language* there are mainly two ways to compute the DBSCAN analysis, which are almost at all similar, but with a slight difference in performances. The function *fpc::dbscan* is the old implementation for the function, and is slow for nowadays standards where there are large and multi dimensional datasets. Instead, the *dbscan::dbscan* function is able to compute almost any dataset in a much more efficient manner, making it the default choice for larger inputs. In our study, we carried on the analysis with the same input parameters and with both the implementations, which gave always the same result. The call to the function only requires the three parameters we already expressed, which are the numerical scaled dataset, the ϵ value and the $MinPts$ value. We proceed now to show the results from this first analysis.

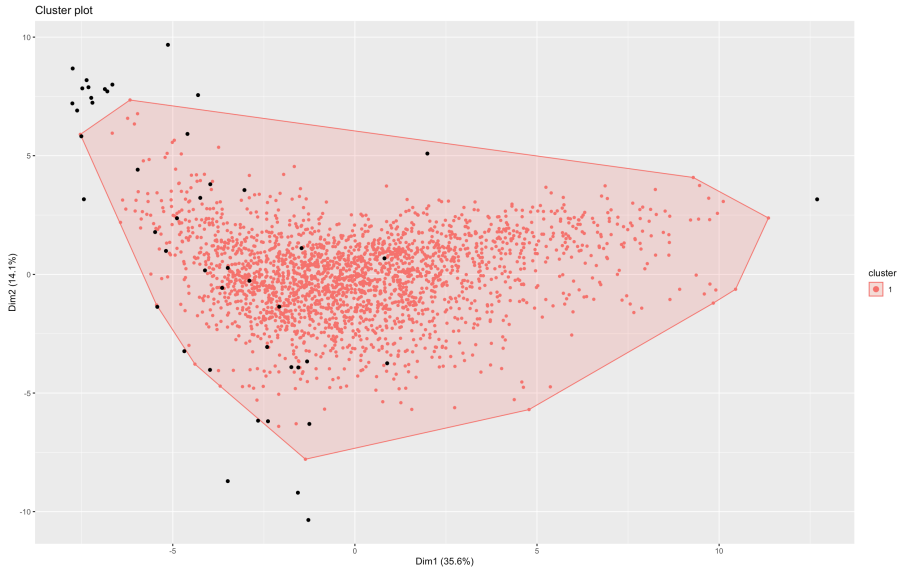


Figure 4.13: Results for DBSCAN execution.

Obviously results from Figure 4.13 cannot be considered satisfactory. The algorithm was only able to find one cluster containing more or less all the observations present in the dataset. We want to note that the representation of the results, based on the first two principal components of the dataset, is just a mean of visualizing, and does not influence in any way the result of the algorithm itself. On why these are the results, many can be the cases, but mainly we can focus ourselves on the followings.

1. The dataset being too overcrowded with similar observations on some statistics. While DBSCAN is indeed useful in real world scenarios, it may be a better fit for data which are not so crowded among the mean in every statistical field. Most players in the NBA cannot either exceed to much or go down a certain treshold in almost every statistical field. In the first case, every player would be a superstar one, in the second, they would be cut by any team. This implies that most of the observations will have particular success in one of the main metrics and statistics, but will inevitably fall in the mean for most of the others.

2. Not being able to give the algorithm the number of clusters. What we discussed in point one, stands as well in the analysis we carried on for centroid clustering, and yet there we were able to see very interesting results. This comes from the idea that centroid clustering allows to choose the number of clusters, which cannot be done for DBSCAN. And while this can be seen as an advantage, in our specific scenario we will see that these are not enough limits, making it impossible to gain any relevant information.

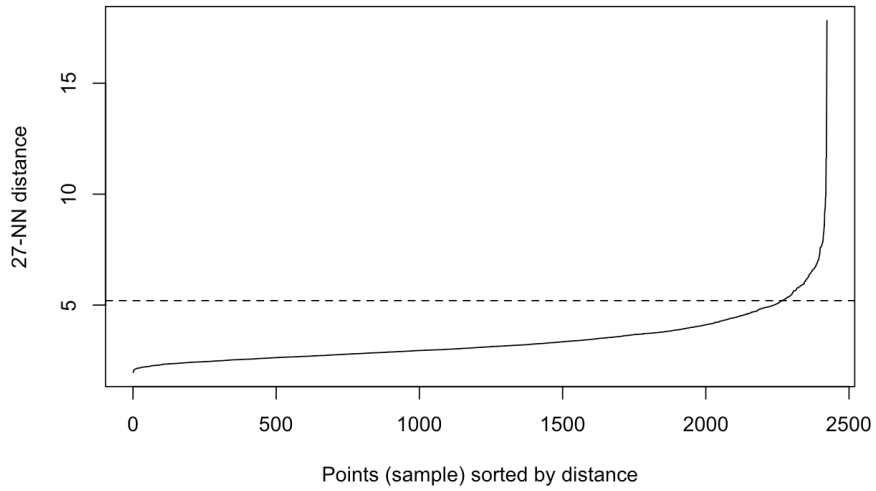


Figure 4.14: K-nearest neighbors distances plotted in ascending order for the reduced dataset.

From these considerations, we are able to gain insight on why the DBSCAN algorithm cannot work on our dataset, as it is, to create a meaningful result. In order to address problem [1] for the previous experiment, we try a different approach, which is based on creating a more slim, and hopefully diverse, dataset. Indeed, some of the information for our dataset can be considered redundant, since it either expresses a defensive proficiency for a player or a superfluous concept. The first case can be seen for variables such as turnovers, steals, or blocks, while the second instance relates to variables such as age, offensive and defensive ratings. We hence remove the following variables from our dataset, to create a subset which highlights variety in the data and relates more to the problem at hand. The removed variables are X, AGE, TO%, eFG%, TS%, SPG, BPG, TOPG, ORTG, DRTG, MIN%, USG%. With this new dataset, we want to try once again the same experiment, repeating the procedure we already analyzed. Firstly, we set $MinPts$ to 15, since we end up with 14 scaled variables after the reduction. Then, we analyze the plot for the distance of the k-nearest neighbors.
While less clear, as Figure 4.14 shows, the elbow is still present, and in particular we assign it to 3. Meaning that the parameters for the execution of the algorithm will be $MinPts = 15$, $\epsilon = 3$.
Unfortunately, our expedient was not enough to produce variability in the dataset, which, as it can be seen in Figure 4.15, is still too much compact around the mean, and is not able to create

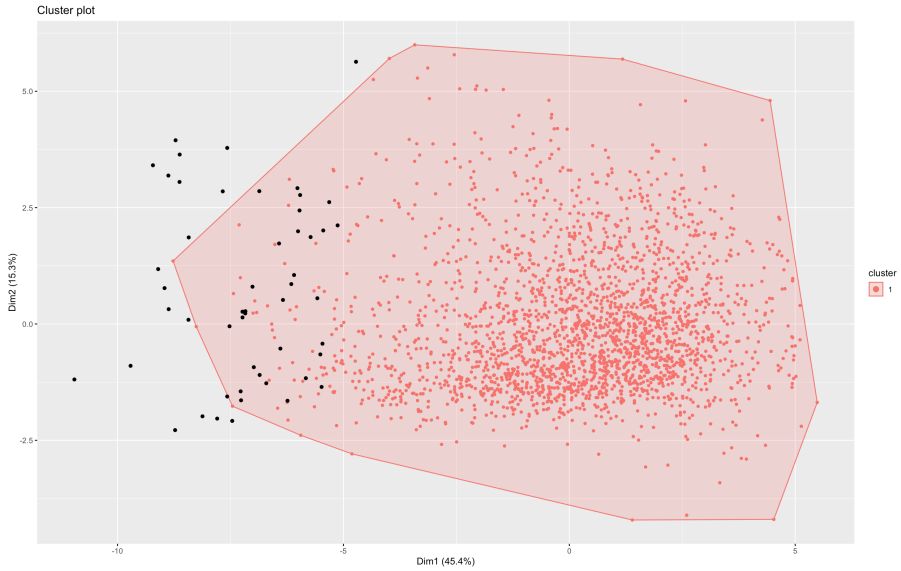


Figure 4.15: Results for DBSCAN execution for the reduced dataset.

more natural shapes. We can conclude, after these experiments, that DBSCAN is not a good solution for the analysis of our dataset as it is.

4.3 Dimensionality reduction - Principal Component Analysis

In this section we apply the principal component analysis, or PCA, to reduce the dimensionality of our dataset. In order to perform the experiments and the analysis on the dataset, the packages used, external from the standard library provided by the *R programming language* and the ones used so far, are *corrr*, *ggcorrplot* and *FactorMineR*.
Since principal component analysis is deeply connected to how much variables are correlated with each other, the first step we will carry on is to look at the correlation matrix for all the covariates present in the dataset.

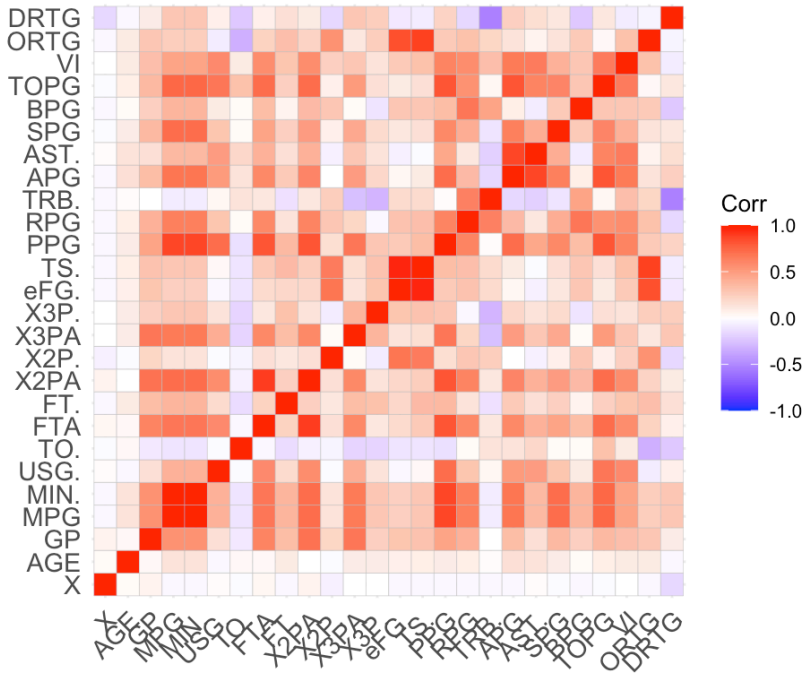


Figure 4.16: Correlation matrix for the whole dataset.

The result from Figure 4.16 is too complex, being it crowded and hence hard to read. We were also able to understand, in Section 4.2.2, that not all the information among the covariates is inherently important for our scopes, meaning we will proceed to analyze, in this section, the subset of variables discussed previously. This way, we can have all the valuable information regarding offensive measurements, allowing a clearer explanation of the results, as well as a facilitation for the dimensionality reduction process. The new correlation matrix will be shaped as in Figure 4.17.

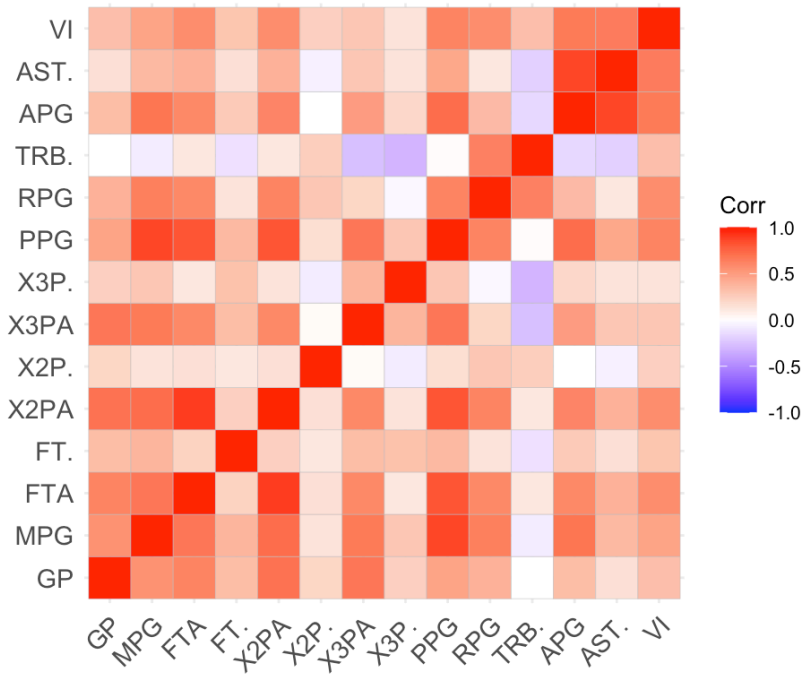


Figure 4.17: Correlation matrix for the reduced dataset.

What we are able to gather from this new result is much clearer, and allows us to assemble interesting considerations on the behavior of players.

- Points per game are mostly influenced by the volume of attempts: a player who shoots, assist or rebound a lot, will likely collect more points. This has a natural connection to minutes per game. A more bold player who can accurately pick his spots will more likely get more play time.
- There are shooters who focus on three point shooting, being 3P% being so deeply correlated with the number of attempts, but also we find some correlation with 2 point shooting and free throw attempts, making them more volatile and complex to guard.
- The TR% for a player has a very low interaction with almost all types of shots, except for 2 point shooting, meaning that we will very unlikely see a whole class of shooting or assisting players who also excel at rebounding.
- Assists are less correlated with 2 point shooting and more with 3PA and 3P%. This is a clear indication of how, over time, point guards cannot be considered anymore traditional, and are instead more and more "going backwards" in the floor, leaving two point shots to bigger, more consistent players. It is indeed more efficient to take a 3 points shot if it has the same probability of going in as a 2 points one.
- The attempts from 2 point shooting are very correlated with free throws, meaning there is an interest in players who can stay inside the area and gather fouls, being physical and playing in the post. Of course there are also more complete players who can shoot both for 2 and 3 points.

- Versatility index is correlated more or less equally with all the main statistical aspects of offensive basketball, which does not come as a surprise. Superstars are defined as such by their ability to excel at most of the aspects of the game.

With this knowledge, we can intuitively know what to expect from the principal component analysis, which can sum up all this information in way less covariates than our original dataset. For our experiment, we will use the function *princomp*, with the following call.

Listing 5 Call for the *princomp* function.

```
res.pca <- princomp(corr.matrix, scores = TRUE)
```

The output for function 5 is an object of class *princomp*, which we can analyze in R via its summary, to gather information about how much variance is explained by each component, and hence how many components are needed for a comprehensive analysis. In this case, 14 principal components were computed, the same number as the remaining covariates. Obviously, we cannot consider them all, since it would defeat the purpose of dimensionality reduction as a whole. We instead notice, by looking at the cumulative proportion section, that by component 4 we are able to explain already 91.34% of all the variance in our dataset. This implies that almost all the dataset can be explained by just using four components. We can see this line of reason graphically, by looking at the scree plot for the resulting object. For this scope, another function belonging to the *factorextra* package was used.

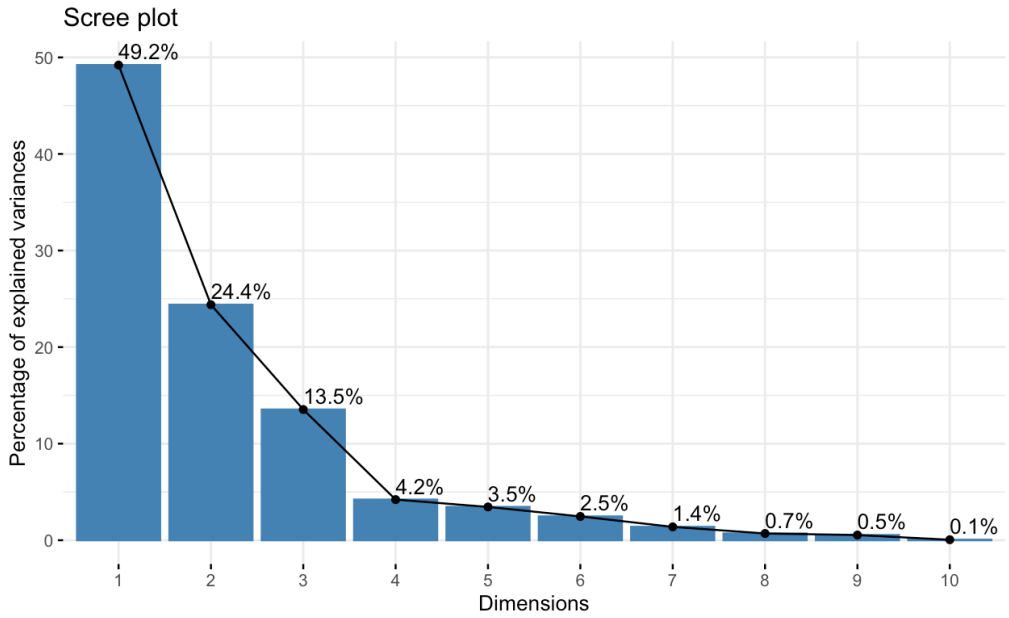


Figure 4.18: Scree plot resulting from the principal component analysis.

Figure 4.18 shows that, by component 4, the largest part of the dataset is explained, and going onward we see that each component explains less and less about our data, meaning we are satisfied with this selection.
Now that we have decided on the number of components, we focus on another topic, which is understanding what these four components explain in the dataset. For each one of them, a loading vector is computed, which shows, intuitively, which variables are explained by each of the components. The problem at hand can be explored by looking at how each covariate relates to to the loading vectors of each principal component. Table 4.5 tries to summarize this, and we can take a look at the results.

Table 4.5: Loading vectors for the four first principal components.

Covariates	Comp.1	Comp.2	Comp.3	Comp.4
GP	0.21861001	0.04989261	0.41191551	0.165400242
MPG	0.33888977	0.11535413	0.14733025	-0.059984948
FTA	0.30786353	0.26768958	0.11295934	0.014530819
FT%	0.10605456	-0.23009869	0.14731534	0.008127742
2PA	0.31359315	0.26450952	0.14619084	0.012837700
2P%	-0.17736815	0.17204186	0.15483731	0.777520530
3PA	0.36054956	-0.11955980	0.29884207	0.059240373
3P%	0.14235527	-0.41952488	0.12761571	-0.397457618
PPG	0.34287963	0.16106356	0.05581184	-0.047437905
RPG	0.02787427	0.48224811	0.09752389	-0.268528854
TRB%	-0.29849835	0.50887304	-0.01188722	-0.306196150
APG	0.37274200	0.02700750	-0.37348524	0.090254569
AST%	0.29623693	-0.04578527	-0.59266378	0.161927136
VI	0.13090892	0.22239491	-0.34599180	-0.010096116

We can see that the components do a good job at explaining at any time almost all the covariates in the reduced dataset, since we decided to highlight in green all the variables positively explained by a component. And while this information could bring us to initial considerations, we would rather like a graphical representation of the table, since otherwise it can be hard to grasp all the novelties in this analysis. We hence opt for a mosaic of biplots. While this may be an unusual concept, having discovered four principal components implying that there may be, combining them, a variety of relationships. These can be discovered via usual biplots, where it is able to underline which variables are influence the most a component, as well as clustering patterns. We decided hence to compute all possible combinations of two dimensional biplots, to get the most complete picture possible.

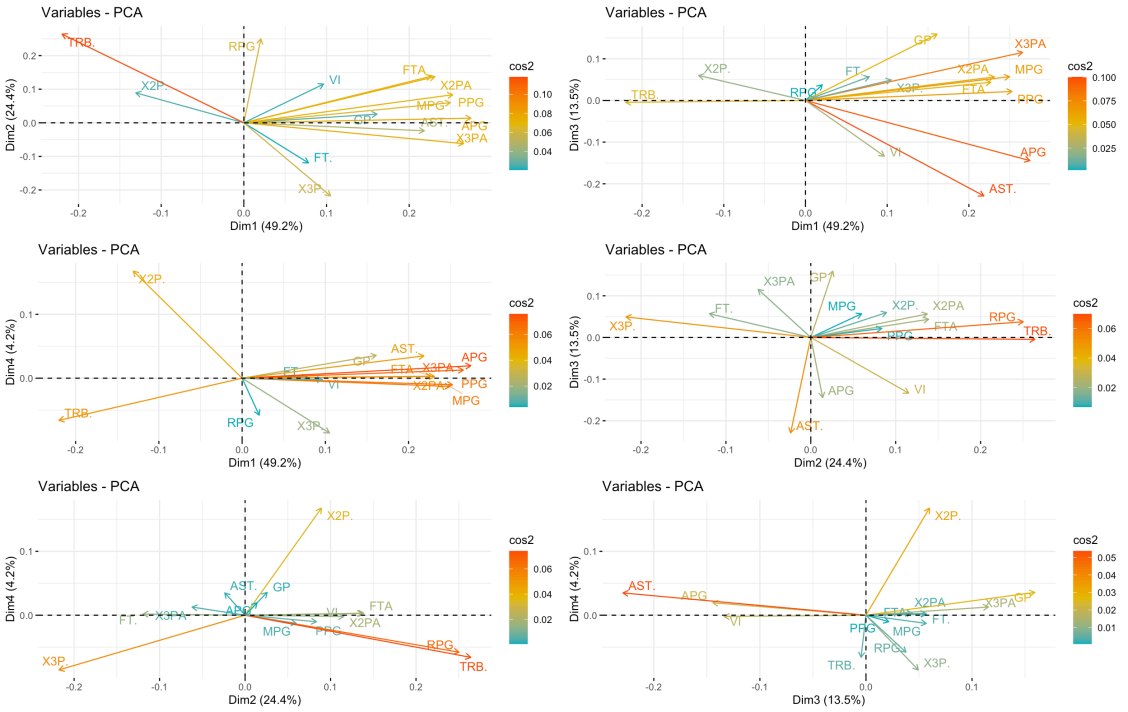


Figure 4.19: Biplots combining all the four principal components, result of PCA.

Firstly, we would like to address the colors used in Figure 4.19. Each variable is indeed colored according to its *cos2* for the considered components. This metric measures the proportion of variance in the variable that is explained by the principal components. It ranges between 0 and 1, and, when close to 1, it means the variable contributes significantly to the variance captured by the principal components and is well aligned with the principal axes. The biplots from Figure 4.19 analyze in details the associations between variables in the discovered principal components: we hope, this way, to find meaningful clusters about types of players, similarly to what we have done in Chapter 2. Going plot from plot, we can gather the following information:

- Components 1 and 2 allows us to see players which are reliable, with lots of minutes per game, and are consistent in FTA, 2PA, 3PA, and PPG. They may well be the superstars of the league. Component 2 also tends to represent pretty accurately the variance referred to rebounds, and we see a smaller cluster of rebounders, with a good value of RPG and TRB%.
- Components 1 and 3 do an especially good job at explaining shooters, where we indeed see good tendencies for 3PA, 2PA, FTA, and, as a consequence, MPG. We see firstly here that dimension 3 has also a big information related to assists, since we see a big emphasys on APG and AST% on the horizontal axys.
- Components 1 and 4 trace the picture for players which can be called ground generals. Indeed, these components represent well modern players with a good value of APG, AST%, 3PA, and 3P%, which are all the metrics which measure a good point guard.

- Components 2 and 3 show a particular influence from RPG and TR%, which means we will more likely see players with big frames who play close to the rim. Indeed, we see that 3P% and AST% influence very negatively these components.
- Components 2 and 4 show players which are hard to pin down. They can be good at two point shooting, while not good at all at 3P%. Indeed, they also show, on dimension 4, a good attitude towards rebounding, making them a good candidate for the "traditional center" category. Still, dimension 4 does not represent particularly well that specialty.
- Components 3 and 4 show another complex scenario to pin down exactly. We see again an influence from AST% and 2P%, but in two very different directions, meaning the covarietes do not couple well together. Also, any other metric is poorly represented, including MPG and PPG, meaning we could be in front of players who still need development.

To conclude our study on principal component analysis, we want to address that this is still a particularly novel approach to such problems, since we weren't able to find any article or even discussion about categorization of basketball players via the use of dimensionality reduction methods. Still, we are satisfied to see a degree of similarity between the results obtained in the k-means analysis, and in the PCA analysis, which implies that, from our dataset, it is indeed possible to gather information about players categorization. Still, by the values of *cos2* in PCA, and dissimilarity in k-means, we can safely say that still a better job can be done on the creation of the dataset, and on the goodness of variables selected. Indeed, for our initial aim, we can safely conclude that the use of advanced metrics, as explained in the beginning of this chapter, could have helped considerably in achieving more meaningful and less trivial clusters. Still we were able to find interesting results, and we will discuss even further how this study can gain more relevance with the use of an advanced and more complex dataset.

5
Data mining methods

This chapter finds its main goal in making predictions about the results found in Chapter 4. We were able to discover a distinction of NBA players in categories, a notion which tries to go further the idea of position, or role. The aforementioned results have been achieved via clustering and dimensionality reduction techniques. We want now to expand more our research, and try to understand how different categories of players can impact the outcome of a game offensively. In practice, this will concern analyzing, and making predictions, about the variable PPG, points per game, for each category of players.
To achieve this result, we want now to give an overview of theoretical aspects and techniques that we are going then to apply in Chapter 6 to the resulting clusters obtained in Section 4.2.1. We want to give a comprehensive list of what we will see in detail before going forward.

- Linear regression
- Principal component regression
- Random forest regression
- Shrinkage methods, in particular Lasso and Ridge regression

5.1 Fundamentals on statistical learning

Data mining techniques are connected to the analysis of existing datasets, with the aim of finding *patterns*. These results are then used, in many realms, to make predictions about future results. The idea of "pattern" in a dataset is, by itself, confusing and vague. In its most general form, it

can be formalized as follows, as for the approach by James, Witten, Hastie, Tibshirani[2] (2021, Chapter 2.1). Given a set of variables known as predictors $X = (X_1, X_2, ..., X_p)$, and a quantitative response variable Y, we want to assess if there is a relationship

$$Y = f(X) + \epsilon.$$

In this case f is an unknown function of variables $X_1, X_2, ..., X_p$, and ϵ is known as an error term, independent of X. At its core, we can hence reformulate the aim of statistical learning as the one of estimating f. Once the estimate of f, known as $\hat{f}$, is found, it can then be used in a new setting

$$\hat{Y} = \hat{f}(X)$$

in order to make predictions for Y. The accuracy level of $\hat{Y}$ actually depends on two factors, namely, the irreducible and reducible errors. Using the best possible statistical learning technique to estimate f will allow us to act on the reducible error for $\hat{f}$. However, even in the best case scenario where $\hat{f} = f$, the irreducible error, introduced by ϵ would still be present. Its nature is related to the fact that f itself cannot measure all aspects of real world data. And if a particular aspect is not measured, f cannot be used to make predictions on it. In the same way as in our dataset we can't infer if, in a particular season, a player faced a debilitating injury, resulting in a drop of his performances. Formally, this can be seen considering the previous relationship $\hat{Y} = \hat{f}(X)$. If we consider fixed $\hat{f}$ and X, we can show that

$$\begin{aligned} \mathrm{E}(Y - \hat{Y})^2 &= \mathrm{E}[f(X) + \epsilon - \hat{f}(X)]^2 \\ &= [f(X) - \hat{f}(X)]^2 + \mathrm{Var}(\epsilon) \end{aligned}$$

where $E(Y - \hat{Y})^2$ represents the expected value of the squared difference between the predicted and actual Y, and $Var(\epsilon)$ instead shows the variance of the error term ϵ. In other terms, we will always have to expect, from our statistical learning techniques, an irreducible error ϵ.
In our treatment of the topic we will range from techniques which are more flexible, and less interpretable, to the opposite. In this sense, we will not always try the absolute best to fit f perfectly. The reason stands in the fact that less flexible models are way more useful when the aim is to infer information on the data. Instead, when the main goals of an analysis are the final predictions, techniques which can adapt more to f while being less interpretable can be preferred. Even in this case, there are risks in fitting too perfectly the target function, with the phenomenon of overfitting. We can discuss this topic in the bigger scenario of evaluation for a model's accuracy. The idea itself of utilizing a wide variety of models comes from the fact that one cannot stand above the others by itself. And the way we assess which model is better than the other is by measuring how well the predictions on the observed data match the latter ones. In the realm of

regressions, the mean square error (MSE) is the most common metric.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\Big\{y_i - \hat{f}(x_i)\Big\}^2$$

Here, $\hat{f}(x_i)$ is the prediction that $\hat{f}$ gives for the ith observation. Obviously, with a value of MSE close to 0 the predictions will be considered accurate. In **??**, we are referring to the training MSE, since it was computed on the training data for a given model. This value though is often considered useless, since every given dataset, before analysis, is splitted into training and testing observations. The first allow the model to be fitted on the function f, while the second ones are used then to assess the quality of $\hat{f}$. In practice, observations known as $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ are used to obtain $\hat{f}$, whereas (x_0, y_0) is a previously unseen test observation not used to train the statistical technique. Hence, we are looking to minimize

$$Ave(y_0 - \hat{f}(x_0))^2$$

known as the average squared prediction error for the test observations (x_0, y_0), or test MSE. Statistical learning holds a particular property, which stands for the most part in all models and datasets: as model flexibility increases, training MSE will decrease, but test MSE may not. And the reason behind it is overfitting. The idea is that the statistical technique is doing too good of a job at finding patterns in the data, picking up some casually generated noise rather than actual properties. This will lead to a large test MSE due to the fact that the model is trying to find patterns that in real data do not exist.
Knowing these few initial fundamentals on statistical learning we can start talk about the techniques we will use in our analysis, starting from the most basic one, being it linear regression.

5.2 Linear regression

Linear regression is one of the simplest and most common statistical learning techniques. Most times, complex studies resort to it as a mean to compute the lower bound of their research. As Ethington Corinna, Thomas Scott and Pike Gary underline in their work[20], this concept was firstly introduced by Sir Francis Galton in 1894, in order to quantify the relationship between variables in a mathematical set. This concept obviously resembles the one of correlation, which enables the research for patterns into data, and then the creation of predictions on future data.
Following the approach from Dastan Hussen Maulud, Adnan Mohsin Abdulazeez[21], in order to understand linear regressions we can classify it in three categories:

- Simple linear regression: a model with a single independent variable, or predictor, X. The dependence is then expressed as $\hat{Y} = \beta_0 + \beta_1 X + \epsilon$.
- Multiple linear regression: the answer variable is predicted using a number of predictors, hence $X = (X_1, X_2, ..., X_p)$. The basic model form for this model is $\hat{Y} = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p + \epsilon$.

- Polynomial regression: a special case of multiple linear regression, in which the each predictor is polynomial, allowing for the creation of curvilinear predictors. The model for this case is expressed as $\hat{Y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2^2 + ... + \beta_p X_p^p + \epsilon$.

Knowing this basic distinction we can talk about the goal of linear regression, which is to fit the best possible values of $(\beta_0, \beta_1, ..., \beta_p)$, coefficients of the model, used to find patterns in the data.
In order to understand how this estimation happens, we can consider the case of a simple linear regression, where we are trying to predict the ith response variable y_i. The estimation will be, in this case, $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$, where $\hat{\beta}_0$, $\hat{\beta}_1$ are our candidates to estimate the optimal values β_0, β_1. Then e_i will represent the ith residual, computed as $e_i = y_i - \hat{y}_i$. Knowing this, we can define the residual sum of squares (RSS) as

$$RSS = e_1^2 + e_2^2 + ... + e_n^2,$$

which we can rewrite as follows.

$$RSS = \sum_{i=1}^{n} \left[y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i) \right]^2.$$

The minimization of this quantity is known as *least square method*, theorized by Gauss in 1809[22], and can be seen in practice in Figure 5.1.

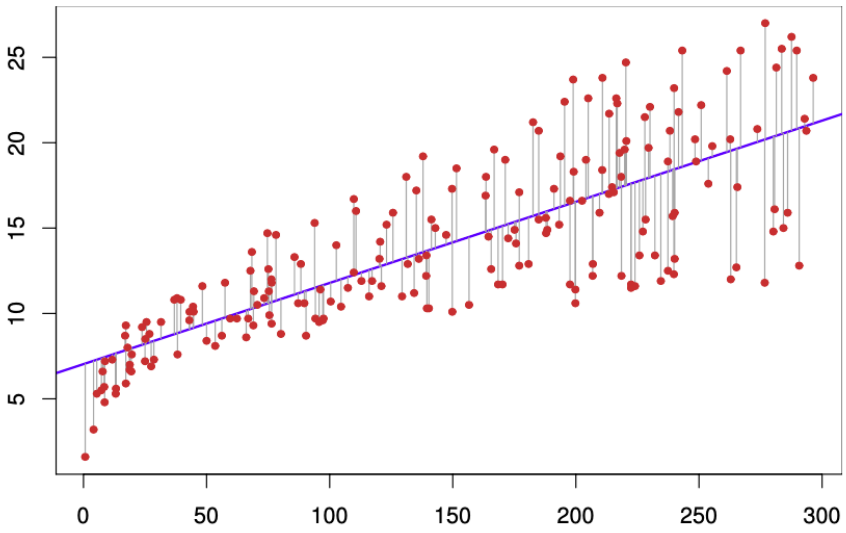


Figure 5.1: Example of the least square method on real data, as for from James, Witten, Hastie, Tibshirani[2] (2021, Chapter 3.1.1). Each grey segment represents a residual.

Once the choice on the coefficients has been done, a fundamental step is to assess their accuracy, especially when there are more than one, as in the multiple linear regression. There are lots of

way to perform such tests, we can, in particular, talk about two of them.
t-Test is used in order to evaluate independent variables, or predictors, $X_i \in X$. It is connected to the concept of hypothesis test, where the aim, in the most general case, is determining whether a variable $x_i \in X$ has a relationship with Y. If this does not hold, it is safe to delete the variable from the dataset, meaning $\beta_i = 0$, since it does not capture any information about the data. Simply put, the t-statistic for a variable can be determined by the following formula for a predictor x_i, where n is the size of the dataset.

$$t_i = \frac{\hat{\beta}_i}{\sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n}}}$$

The resulting value is connected to a Student-t distribution, which is then used to determine the probability of seeing any number greater than or equal to $|t|$ under the premise that $\beta_i = 0$. A tiny result suggests that it is unlikely to see such a relationship between the predictor and the response through random chance. This probability is known as the *p-value*. In this situation, X_i is retained for analysis. The variable is eliminated if the opposite is true and the p-value is high..
The considerations on p-value are not always stable, and hence, for completeness, we briefly discuss the *F-test*, which is more stable in the context of multiple linear regression. This test considers a bigger hypotheses test, where the whole regression coefficients are investigated on whether or not they are 0. The F-statistic is computed for this test as follows,

$$F = \frac{(TSS - RSS)/p}{RSS/(n - p - 1)}$$

where TSS is the total sum of squared differences between each data point and the mean of the dependent variable. The numerator computes the so called mean square for the model containing all the covariates we are analyzing for the hypothesis test, while in the denominator we are instead considering the mean square for a model with only the intercept β_0. The numerator difference is then divided by the degrees of freedom for the model with all the covariates. This number corresponds to the number of predictors. Instead, the degrees of freedom for the denominator is equal to the total number of observations, minus the number of predictors, minus one. The results are easy to interpret: when the final computed value is close to 1 we have a strong case for discarding all the coefficients in the analysis. If, instead, the F-statistic is larger than 1 we can reject the hypothesis and keep the selection.
In the realm of multiple linear regression, once the analysis of the F-test first, and the p-values then are done, it is important to decide which are the most important variables in a predictor. Most times the response variable is associated with only a subset of predictors, and its formation is called variable selection. This can be done of course by hand, iteratively removing one at the time the least important variable until a satisfactory model is obtained. But in the case where p, the number of predictors, gets large this process could become unbearable. For this reason, three methods are generally used, in order to make this process systematic and automatic.

- Forward selection: it starts with an empty model, with no predictors and just the intercept β_0. Then, p linear regression models are fitted, and the variable which has the lowest RSS

is added to the empty model. Until a breaking condition is reached, or the whole variables are added, this process continue.

- Backward selection: the starting point is a model with all the variables, and at each iteration the one with highest p-value is removed. This process is done until a breaking condition is reached, or all the variables are removed.
- Mixed selection: a combination of both forward and backward selection, where we start with an empty model as in the first case. Then, the variable which acts as the best fit is added, and this process is repeated. If, at some point, one of the variables reaches a p-value too high, it gets discarded from the model. This process is continued until all the variables present in the model have a reasonable p-value, and all the discarded ones have a p-value too high if added to the model.

Forward selection, among all, is the considered a greedy approach, since it might include predictors which are early on considered meaningful, and going onward become less and less important. This issue is fixed by mixed selection.

5.3 Principal Component Regression

Principal Component Regression finds its fundamental concepts in two notions, being dimensionality reduction, described in Section 3.3, and the creation of principal components starting from a dataset, discussed in Section 3.3.1. We will not discuss again these concepts, but they are the basis for what we are going to see now. The only difference that stands, in this scenario, is the aim, which in this case resides in building a regression model rather than a clustering one.
The key idea is similar to the one we have already seen for clustering: once $Z_1, ..., Z_M$, with $M \leq p$, principal components are computed on the original dataset, we want to obtain the smallest set of them which is able to capture most of the variability in the data. Usually, what happens is that the directions in which $X_1, ..., X_p$ show the most variation are the directions associated the most with Y. This is the fundamental assumption for Principal Component Regression. If this holds, as it usually does, it is obviously more convenient to fit a least squares model on $Z_1, ..., Z_M$, rather than on $X_1, ..., X_p$, since most of the information on the data is the one captured by the first set. An obvious first advantage for this method is avoiding overfitting: we are losing, creating M principal components, some information which may not relate to the response variable, and which risks to be recognized and inserted in the regression model. It is also easy to see that it can reduce the variability of a dataset, summing up the content of the predictors. And while this is not a feature selection method, as the automatic ones we have seen at the end of Section 5.2, it can still be seen as an improvement. Indeed, we are not leaving behind any predictor, and we are giving value only to the ones which actually matter. Still, what PCA uses is a linear model. It implies that the least squares can actually steer the analysis in the exact opposite direction of progress, if a dataset has a hidden nonlinear pattern. Also, Jiang Hong and Kent Eskridge[23] found out that the bias of PCA results was affected by the sampling error in their trials, meaning that this method still have flaws and has to be treated in a way to minimize them in order to function efficiently.

5.4 Random Forest Regression

In order to first understand how random forest regression works, some preliminary theory pieces have to be addressed, as they are the basis for the model we are going to see. In particular, we are referring to regression trees and bootstrap sampling, also called bagging.
Regression trees can be considered as a variant of decision trees, designed to approximate values related to real world scenarios. A decision tree is generated when each decision node in the tree contains a test on some input variable's value. This can be achieved via what's called binary recursive partitioning, as explained by Hastie, Tibshirani, and Friedman[24]. It is an iterative process that splits the data into partitions or branches, and then continues splitting each partition into smaller groups as the method moves up each branch. In practice, it works as follows: initially, all records in the Training Set are grouped into the same partition. The algorithm then begins dividing the data into the first two partitions, or branches, using every possible binary split on every field. The algorithm selects the split that minimizes the sum of the squared deviations from the mean in the two separate partitions. This splitting rule is then applied to each of the new generated nodes. The step of dividing is then repeated until each node reaches a user-specified minimum node size and becomes a terminal node. If the sum of squared deviations from the mean in a node is zero, then that node is considered a terminal node even if it has not reached the minimum size. It is easy to see that this process inherently generates overfitting in the model: the tree will try to explain every single deviation in the training set, ignoring that one of them may be generated from instability of real-world data. This creates a scenario with very high variance. The other principle we have to explain before going further with explaining random forest regression is bootstrap sampling, also called *bagging*.

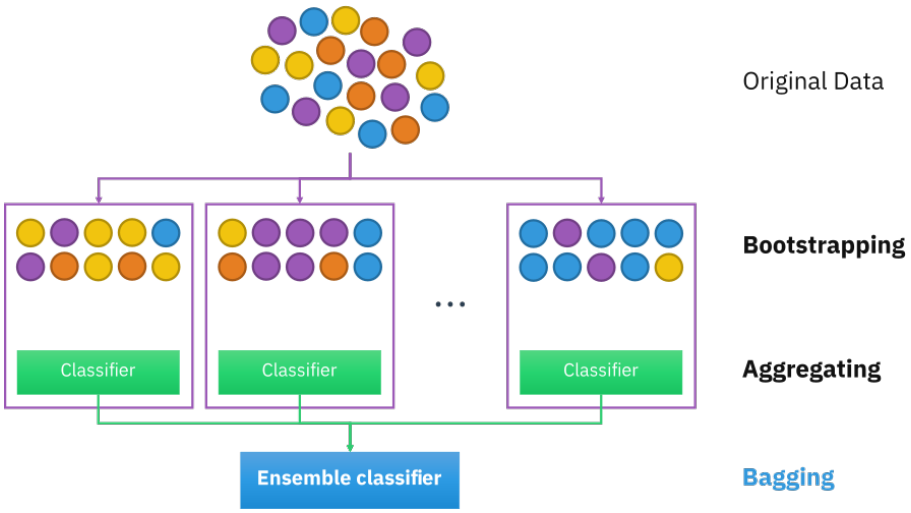


Figure 5.2: An illustration for the concept of bootstrap aggregation, By Sirakorn - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=85888768`.

This technique tries to address the instability and overfitting issues related to regression trees. In particular, the technique, which can be visualized in Figure 5.2, works as follows. Given a standard training set X of size n, bagging generates m new training sets X_i, each of size n', by sampling from X uniformly and with replacement. By sampling with replacement, we intend that some observations may be repeated in each X_i. When $n' = n$, each set X_i is expected to have $1 - \frac{1}{e}$, or roughly 63.2%, of unique examples from X, the rest being duplicates, as proven by Aslam, Javed, Popa, Raluca [25]. This kind of sample is known as a bootstrap sample. Sampling with replacement ensures each bootstrap is independent from the others, as it does not depend on previous chosen samples. Finally, m models are fitted using the above m bootstrap samples. The final results are combined either by averaging the output, in the case of regression models,

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x)$$

or by voting, for classification ones. While this method indeed reduces the variance introduced by decision trees, it is expanded even further by the random forest technique. The reduction of variance is given by the concept of averaging the results of multiple trees using not correlated datasets. Bootstrap sampling is hence a way of de-correlating the trees by showing them different training sets. Finally, it is possible to compute the measure of uncertainty in the prediction.

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x) - \hat{f})^2}{B-1}}.$$

This value is obviously deeply influenced by the number of samplings B, which can be decided thanks to cross-validation. Other measures for this estimating the error rate for bagging can be, for example, the Out-of-Bag estimation.

It is finally possible to move from bagging to random forests regression methods. The general method of random decision forests was first proposed by Tin Kam Ho in 1995 [26]. The small tweak that set random forests away from bagging is in a further decorrelation of trees. The procedure remains the same, in the sense that a number n of trees are built on bootstrapped training samples. During the construction of these trees, what happens is that a random sample of typically $m \approx \sqrt{p}$ predictors is chosen as split candidates, where p is the whole set of predictors. The regression model is allowed to use only one of the proposed splits. In other words, the model used for regression in a random forest is not allowed to consider the whole set of observations, nor the whole set of predictors.

The rational concept behind this choice is pretty intuitive: suppose that, in a regression scenario, there is a predictor that is clearly stronger than the others. By using only bagging, it is highly possible that all the generated trees will look like one another, since they will all base themselves on the specific strong predictor. Hence, the predictions from the bagged trees will be highly correlated. In this worst case scenario we described, it may not lead to a substantial reduction of variance with respect to the use of a single decision tree. In this sense, the idea of regression forests is to suppress this issue by allowing each model to consider only a set of predictors. On

average, in the worst case, $\frac{p-m}{p}$ of the splits won't consider the strong predictor, leaving the other a fair field. This is why random forests are used in the context of decorrelating trees. A large issue in random forest is in the choice for the size of m. When $m = p$, we are simply applying bagging, while $m = \sqrt{p}$ may not always be the solution. While this remains an issue, once again applying a cross-validation approach may solve this issue. Instead, the main problem relating random forests is in their interpretability. By far the best advantage of simple models such as linear regression or decision trees is in their interpretability: being able, for a developer or an analyst, to understand easily why a model is acting in a determined way. Random forests suppress this advantage, in favor of minimizing variance and bias in the final predictions.

5.5 Shrinkage methods

We analyzed, at the end of Section 5.2, techniques and methods which are commonly used when the aim is to simplify a model. In that particular case, a subset of the original variables is chosen, based on their influence on the final model. Later, in Section 5.3, we were able instead to sum up the content of all variables into a number of principal components. The aim is clear and simple so far: we want to achieve a model that finds pattern in the original data with a low MSE, while avoiding to overfit those data. The so called shrinkage methods try a different approach from the ones seen so far. Instead of summing up the content of all the variables, or excluding the non meaningful ones, the aim is to constraint or regularize the estimated coefficients $\hat{\beta}_1, ..., \hat{\beta}_p$ associated to the p predictors of the model. The two examples we will see are ridge regression and lasso, which are the fundamentals for these kind of techniques.

5.5.1 Ridge regression

The concept nowadays known as ridge regression was thought and invented in many different contexts throughout the beginning of the XXth century. In particular, in the 1920's, it began widely known thanks to the work of Andrey Tikhonov[27] and David L. Phillips. The main concept is pretty straight-forward. It shares lots of similarities with the least squares, except the coefficients are estimated by minimizing a different quantity. In particular, we are talking about

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

which can be rewritten as

$$\hat{\beta}_{\text{ridge}} = \min_{\beta} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$
$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 \leq t.$$

where $\lambda \geq 0$ is called a tuning parameter, which is separately found, generally via cross-validation. The second term of the first equation is called shrinkage penalty, and it is a small quantity when $\beta_1, ..., \beta_p$ are close to zero, with the effect of shrinking these estimates close to zero. In the particular case where $\lambda = 0$ no term is penalized and the resulting model is the same as the one obtained with traditional least squares. On the other hand, where $\lambda \to \infty$, the impact of the shrinkage penalty grows too much, and the coefficient estimates approach zero. The output of ridge regression will be different with respect to the value of λ. Of course we will everytime see an approximation for $\beta_1, ..., \beta_p$, but a different set will be produced for each value of λ, hence $\hat{\beta}_{\lambda}^{\text{ridge}}$. We want now to address the advantages of ridge regression, where the biggest one stands in the bias-variance trade-off. With an increase of λ, the flexibility of the ridge regression fit decreases, leading to consequent decrease in variance and an increase of bias. Then, with $\lambda = 0$, the variance is high but there is no bias. The key to ridge regression is hence finding the good fit for λ, which can be chosen by analyzing the trend for MSE with respect to the value of the tuning parameter itself. When λ reaches its optimal value, most times going further will not give any more significant decrease for the MSE. Ridge regression has its best utility in scenarios where the results for linear regression found a very high MSE. This recalls the concept explained at the beginning of Section 5.2, for which linear regression was a lower bound with respect to the optimal result. Obviously, ridge regression has an obvious advantage over subset selection efficiency, since the latter has to analyze 2^p models, whereas, for every single value of λ, ridge regression fits a single model.
The main flaw with ridge regression is clear and easy to see. Unlike the methods explained at the end of Section 5.2, Ridge Regression will, at the end of the of its process, include all the predictors in the final model. The penalty factor $\lambda \sum \beta_j^2$ will shrink all coefficients *towards* zero but never *exactly* to zero. And while this issue does not constitute a real issue in terms of prediction accuracy, it can lead to a lack of interpretability when p, the number of predictors, gets large.

5.5.2 Lasso

In the realm of statistical learning, lasso, or least absolute shrinkage and selection operator, is used to perform both variable selection and regularization, in order to enhance both prediction accuracy and interpretability. The term and its formalization were popularized by Robert Tibshirani[28].
The lasso estimation for the coefficients is pretty similar to the ones computed by ridge regression,

and can be expressed as follows

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|=RSS+\lambda\sum_{j=1}^{p}|\beta_j|$$

which can be rewritten as

$$\hat{\beta}_{\text{lasso}}=\min_{\beta}\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{p}x_{ij}\beta_j\right)^2$$
$$\text{subject to }\sum_{j=1}^{p}|\beta_j|\leq t,$$

where, as it can be seen, the penalization factor includes a norm of the coefficients, with respect to ridge regression. Lasso behaves similarly to ridge regression in the sense that shrinks some coefficient estimates towards zero. But in the case of this new method, the penalization has the effect of forcing some of the coefficient estimates to be exactly zero when the tuning parameter λ is sufficiently large. Indeed, when lasso is performed we are trying to find the set of coefficient estimates that lead to the smallest RSS, as in the least squares, but with the constriction that there is a response variable t for how large $\sum_{j=1}^{p}|\beta_j|$ can be. If the response variable is pretty large, then the constraint does not have to be too much restrictive, and the coefficient estimate can be closer to the least squares. If instead t is small, then the quantity must be small in order to avoid violating the constraint $\sum_{j=1}^{p}|\beta_j|\leq t$. Obviously this line of reason apply similarly to ridge regression, with the difference that, in this case, some of the coefficients are allowed to be exactly zero.

As in the ridge regression, but with an even higher importance, finding the correct tuning parameter λ is extremely important. In this case, the risk is to not consider some parameters which can indeed be meaningful for the analysis, as in the case where λ is too large. Again though, a good selection of this parameter can be done through cross-validation.

We can conclude this treatment by considering whether or not there is a clear better candidate between ridge regression and lasso. The analysis of examples in the literature, in particular from James, Witten, Hastie, Tibshirani[2] (2021, Chapter 6.2.2, Figure 6.9), show that there isn't a method which always overcome the other. In general, the line of reason is again simple: in cases there are models where the predictors highlighted from the least squares are substantial, lasso is preferred due its ability to shrink some of them even to zero. Instead, ridge regression will perform better when the estimated coefficients are of roughly equal size. We recall once again that the best way to find the tuning parameter, as well as the method which provides the least MSE is the cross-validation. Finally, in general lasso can be employed in situations where better interpretability is preferred, while ridge presents more variability, in contexts where predictions are the clear preference.

6

Data mining predictions

The primary objective of this chapter is to implement the data mining methodologies outlined in Chapter 5 on the specific issue at hand. In Chapter 4, we established that it is indeed possible to categorize NBA players effectively by analyzing their performance metrics, as indicated by their statistical records during a given regular season. Building upon this foundation, our current focus is to employ these data mining techniques for each newly identified category. By doing so, we aim to check the possibility of deriving meaningful predictions within each cluster. Moreover, our intention is to extract insights that can guide improvements in order to enhance player performances.

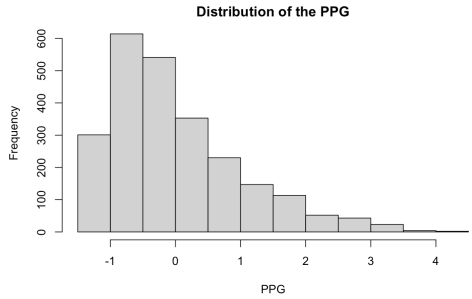
6.1 Preparation of the dataset for analysis

As a first step, we need to load the results obtained in Chapter 4. Among all the ones we obtained, we chose to consider in particular the ones from Section 4.2.1, being them the 9 clusters resulted from k-means analysis with Hartigan-Wong. This choice is motivated by the relative homogeneity across the various k-means results. In essence, the selection of this specific result does not anticipate significant deviations in the subsequent analysis, making it a suitable candidate for our investigation. Moreover, this result comes in a particularly handy format, being it 9 subsets of the original dataset, which means it is much easier to manipulate for an analysis, with respect to the results of, for example, PCA, where 4 components were used to sum up the predictors. In a similar way, we opted for not excluding any predictor from the dataset, even the ones which are counter intuitive to keep, such as SPG, BPG, or DRTG. We just removed, for obvious reasons, POS, TEAM and FULL.NAME, being them the only categorical variables in our initial dataset. The obtained datasets were then scaled, since it fosters fairness, optimization, and robustness across a

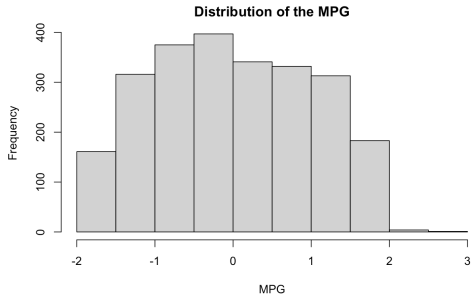
spectrum of techniques, such as the ones described in Chapter 5. This preparatory step ensures that the subsequent analyses are reliable, unbiased and easily interpretable. Furthermore, we opted for creating, in each subset of the original dataset, a separation between training and testing observations. The partitioning of data into training and testing sets fundamentally addresses the requirement to measure the model's ability to generalize beyond the data it was trained on. When a model is trained solely on a specific dataset, there is a risk of overfitting, whereby the model might memorize the training data rather than truly understanding the underlying patterns. This would result in poor performance when presented with new, unseen data. For each partition, we opted for a separation where $\frac{2}{3}$ of the observations are used for training, and the remaining $\frac{1}{3}$ is used for testing the model. The distribution of a larger portion for training leaves to the model the opportunity to discern complex patterns and relationships within the dataset. This exposure assists the model in capturing patterns and generalizing trends effectively.
Finally, before moving forward, we want to address the topic of the chosen response variable. Indeed, there isn't in our scenario a clear candidate for this role. Our final aim is to measure whether it is possible to make predictions on the offensive proficiency of players given their category, which is rooted on the idea that each category of players will excel in different aspects of the game. Hence, we focus ourselves on the three main metrics which measure offensive effectivness of a player.

- PPG: a direct and intuitive metric that quantifies a player's scoring contribution. It reflects the average number of points a player scores in each game they participate in. PPG is valuable as it encapsulates a player's ability to consistently contribute to their team's offensive output.
- MPG: this metric is crucial because it accounts for a player's playing time, which directly influences their opportunity to contribute offensively and participate to plays on the court. Players who have higher MPG are entrusted with more playing time to impact the game, indicating their value to the team's offensive strategy and execution.
- ORTG: it is an advanced metric that evaluates a player's offensive efficiency. It represents the number of points a player produces per 100 possessions while on the court. ORTG goes beyond raw statistics by factoring in a player's scoring contribution in relation to the team's overall offensive possessions. It is more complex, since it contains aspects like scoring, passing, and overall offensive decision-making.

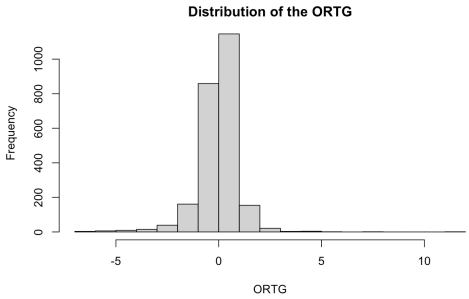
Each of these metrics is a good candidate, with obvious pros and cons, which can be further elaborated by looking at their distributions as in Figure 6.1. First and foremost, we can see that the PPG distribution has a right-skewed normal distribution, whereas the MPG roughly has no tails, and has most of the observations around the mean of the curve, and finally ORTG suffers a similar problem. Similarly to MPG, most of the observations position around the mean, making it difficult to assess any real information, but it presents a few players on the tails. Basing ourselves on the description we made earlier, it is easy to understand why this is happening. Most players will play a similar number of minutes per game, and indeed the distribution creates a sort of categorization: there are players who will play few minutes per game, most players who will play around the mean of the league during a game, and a few superstars who will be held responsible



(a) Distribution of PPG.



(b) Distribution of MPG.



(c) Distribution of ORTG.

Figure 6.1: Comparison of candidate response variables distributions.

on the floor almost at all times, increasing considerably their MPG. In ORTG we can define a similar pattern, where most players will position themselves around the mean of the league, and there are very few players which are below and above. In practice, a response variable with most observations around the mean can lead to the construction of a sub-optimal model by limiting variability, biasing coefficient estimates, inflating p-values, hindering generalization, and increasing the risk of overfitting. It's crucial to have a diverse range of response values to allow the model to accurately capture the true relationships between predictors and the response. For this reasons, and given the fact that it is the most straight forward metric, we will choose PPG as the response variable for our analysis. We have highlighted that this does not follow a normal distribution, and we furthermore want to address that the canonical transformations, logarithmic or power, did not help this issue. Not having a normal distribution for a response variable can introduce potential problems, related to assumptions of statistical methods and models. While these issues might be slight in some cases, we will always try to understand if this restriction is being too limiting for the model at hand. With all the preparations for the dataset finished, we can start addressing the resulting models for each technique. In every analysis, 9 models were computed, due to the presence of 9 subsets of the original dataset. This process was entirely automatized via the *R programming language*, and additional libraries needed for each analysis.

6.2 Linear Regression

In our current problem, the application of traditional linear regression presents a significant challenge which originates from the dimensionality of the dataset. With a substantial 25 covariates involved, and 9 clusters to analyze, the resulting number of potential models becomes overwhelmingly vast, $2^p \cdot 9 = 301989888$, rendering thorough analysis practically infeasible. Traditional linear regression techniques entail estimating coefficients for each covariate, and with such a multitude of predictors, the model complexity explodes exponentially. This complexity not only hampers the interpretability of the results but also intensifies the risk of overfitting, wherein the model may capture noise rather than meaningful patterns. The computational burden of processing an enormous number of models further intensify the issue. Given these considerations, the use of traditional linear regression becomes impractical and consequently we rely on automatic selection methods, described in Section 5.2.
The setup for the experiments is the same in all the three automatic selection methods applied, which are forward, backward, and mixed selection. We can explain the process as follows.

1. At the beginning, 9 automatic selection models are trained on the training data of each cluster, utilizing the *regsubsets* function, from the *leaps* library.
2. For each of the computed models, we store three metrics, which are utilized to understand the best subset of variables. These are the RSS, Adjusted R^2, and BIC. Since the RSS always suggested to utilize all the original variables for the analysis, we decided to consider only the latter two metrics.
3. Two models are computed in this step, one which contains the subset of variables suggested by the Adjusted R^2, one with the ones suggested by the BIC, utilizing once again the training data of each cluster.
4. To understand which of the two newly computed models is better for each cluster, the function *anova* is applied, since, in each case, the model suggested via BIC is contained by the one obtained via Adjusted R^2. Then, a threshold of 0.10 is utilized to understand the results of the function anova. If the p-value obtained from the function is smaller than the treshold, then the hyphotesys test is confirmed, and we can keep the more complex model, the one suggested by Ajusted R^2, otherwise we keep the simpler one, suggested by BIC.
5. The residuals for each of the final chosen models are plotted, as well as the predictions done on the testing data.

Doing this process, we are able to cut the complex number of models we need to analyze from 301989888 to just 27, since for each cluster three automatic selection methods are applied. These are still too many for the purposes or our analysis, and hence we try to further reduce this number, with the aim of going down to a single model for each cluster. This is done, again, by analyzing statistics of the different models. For each of the three models computed, we analyze the RSS, or residual sum of squares, the Adjusted R^2, and the F-statistic. The models which achieve, among the three, the least RSS, and highest Adjusted R^2 and F-statistic are the ones we will show in the results of this analysis. This analysis gives us another opportunity, that being to analyze which of the automatic selection methods is better for our case study. Table 6.1 shows us the results of

the aforementioned analysis.

Table 6.1: Comparison of the results from the automatic selection methods.

	RSS	Adjusted R^2	F-statistic
1 (M)	0.201	0.962	349.67
2 (B)	0.213	0.953	49.97
3 (M)	0.207	0.955	495.89
4 (B)	0.108	0.988	515.79
5 (F)	0.121	0.984	694.06
6 (F)	0.273	0.925	218.55
7 (M)	0.264	0.918	290.46
8 (B)	0.171	0.969	506.03
9 (M)	0.136	0.981	713.34

Beside every cluster's number, we highlighted the model which gave the best overall results. The results seen here are very encouraging for the scopes of our analysis. We are able to underline a low overall value for the RSS, as well as values for the Adjusted R^2 which never go below 0.90, which is considered a safe threshold, and finally, the F-statistic value is always high enough to confirm that we are not considering a subset of variables without any significance. Knowing all of these results, and the methodologies of our analysis, we can go on showing the results. For each cluster, only the best model's residuals and predictions will be shown, going from the first to the last cluster.

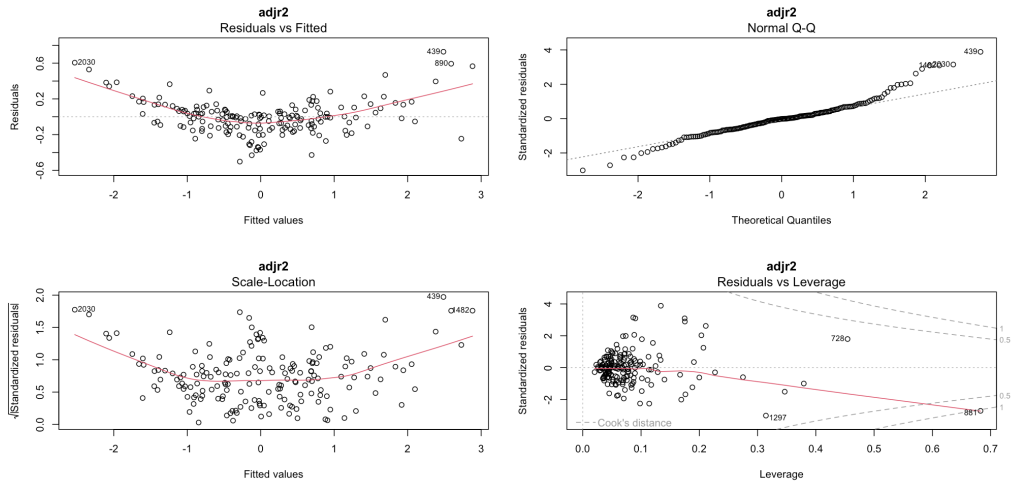


Figure 6.2: Residuals for cluster 1 with linear regression.

Figure 6.2 shows the residuals obtained by the mixed selection approach for cluster 1. We can consider these results as somewhat satisfactory, considering that, apart from the "Residuals vs

Fitted" plot, which resembles on some sort of shape, following the red line. This does not happen in the "Scale-location" plot, which shows a cloud of points. The "Normal Quantile-Quantile" is extremely satisfactory, showing almost all points on line, and the final "Residual vs Leverage" plot shows a few points which Cook's Distance is close to 1, meaning they can be identified as outliers. As we have seen in Chapter 4, these are expected in our scenario, due to the nature of our data. Figure 6.3 show the predictions for this model.

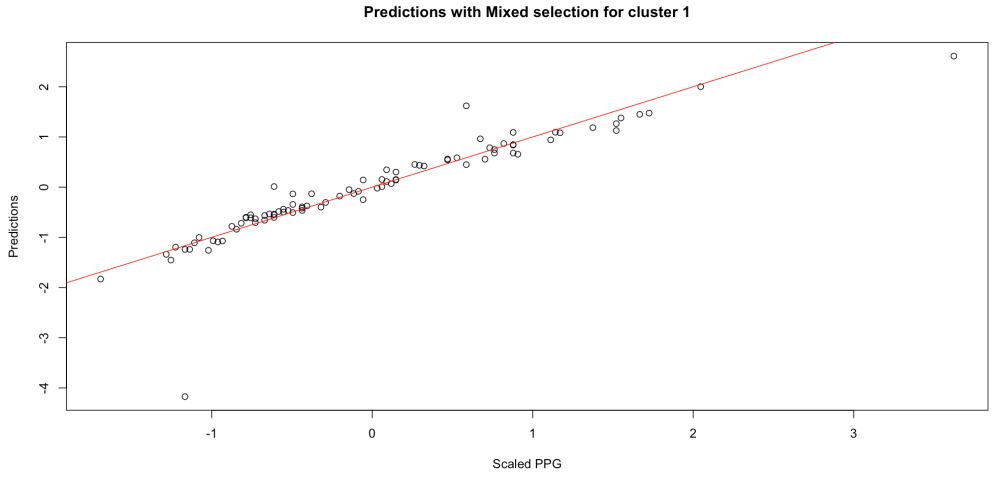


Figure 6.3: Predictions for cluster 1 with linear regression.

Just by looking at these results we can consider ourselves partially satisfied. This is by far the easiest model used for predictions, and it is already showing most points on the red bisector, or at most close to it, which is an interesting starting point. The point to the furthest right is the outlier which was also highlighted by Figure 6.2 most certainly. The variables selected by this model, for the cluster relating the so called "ground generals" are 14, with a particular weight on MPG, USG%, 2P% and APG. It shows us that the best way that guards have to influence the offensive end of a game is by creating assists or two point attempts. This last particular information reveals us that, most probably, these players will leave more complex three point shots to more specialized players in that department.
The results for cluster 2 are not shown since are not deemed as satisfactory on a graphical standpoint. The main reason for why this applies only to this subset of players is related to the number of observations we find in the correspondent cluster. There are only 47 of them, which implies that 12 were used for the predictions, and the remaining 35 for the training of the model. And while it could be interesting if this was from the ground up a meaningful category, we would like to recall that this are the players who are most likely to get next to none minutes on the floor, and probably relegated to a lower level of basketball play. To evaluate their performances, it would be more meaningful to have them included in a larger dataset with players of their lever, to understand the flaws and variety of their game. For these reasons, we move to cluster 3, which sees presence of smaller sized shooters with less minutes than the average players of the league.

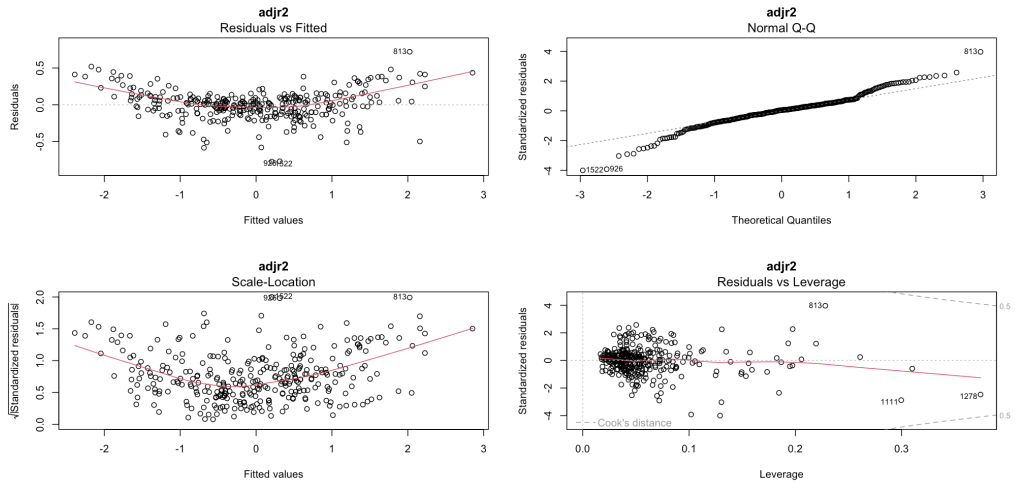


Figure 6.4: Residuals for cluster 3 with linear regression.

Figure 6.4 are similar to the ones seen before, but while the "Residual vs Fitted" gets better, likely due to the presence of more observations in the cluster, we can see that the "Normal Quantile-Quantile" gets worse, due to a deviation in the final part of the line. Finally, in this case, we see that there are no players over the level of 1 in Cook's distance, meaning there will not be any significant outliers.

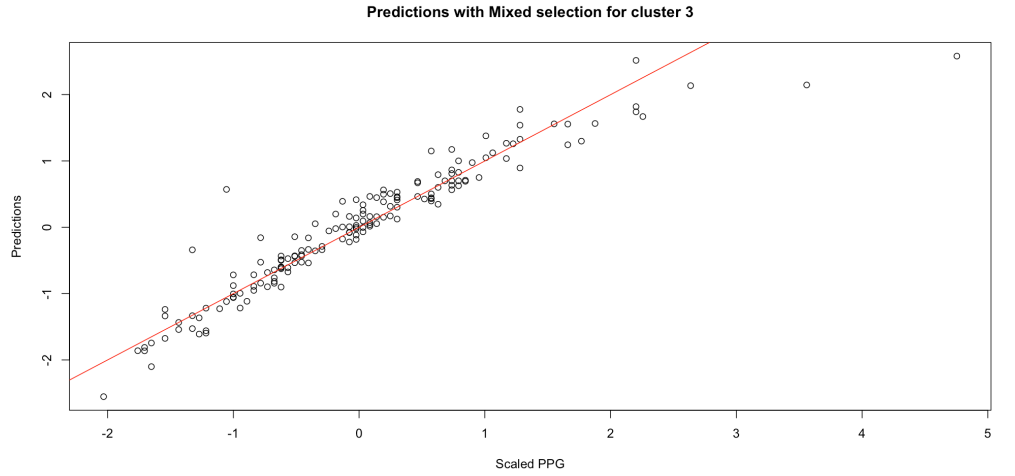


Figure 6.5: Residuals for cluster 3 with linear regression.

In Figure 6.5 we can see a significant presence of observations which are on the bisector, or close to it, as well as points which deviates a lot, showing a high error. This can be due to a reason: the cluster considered for this analysis contains players which are in a sort of gray area, since they are

not really relied on from their team, but are expected to contribute to the offense of a game. For this reason, it is possible to see players who can play few games and have a great performances, as well as ones who show up lots of times in a season with a contained number of PPG. This may be the reason why some observations are harder to pin down than others. Indeed, the coefficients underline that the variable which most influences their PPG is the minutes they play in a game. Getting more minutes will naturally lead to more points for these players, and they can be scored in a variety of ways, especially 2PA, and 3PA, which are coefficients that positively influence the predictions.

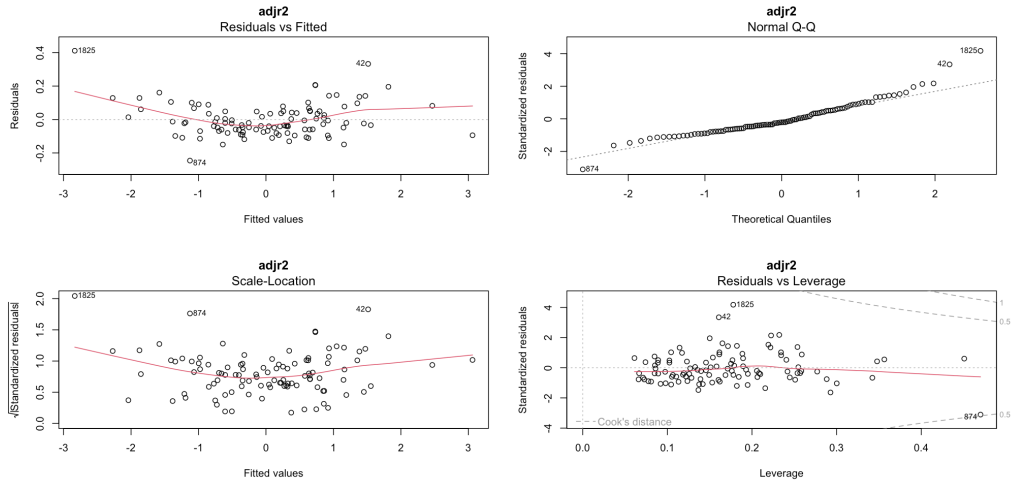


Figure 6.6: Residuals for cluster 4 with linear regression.

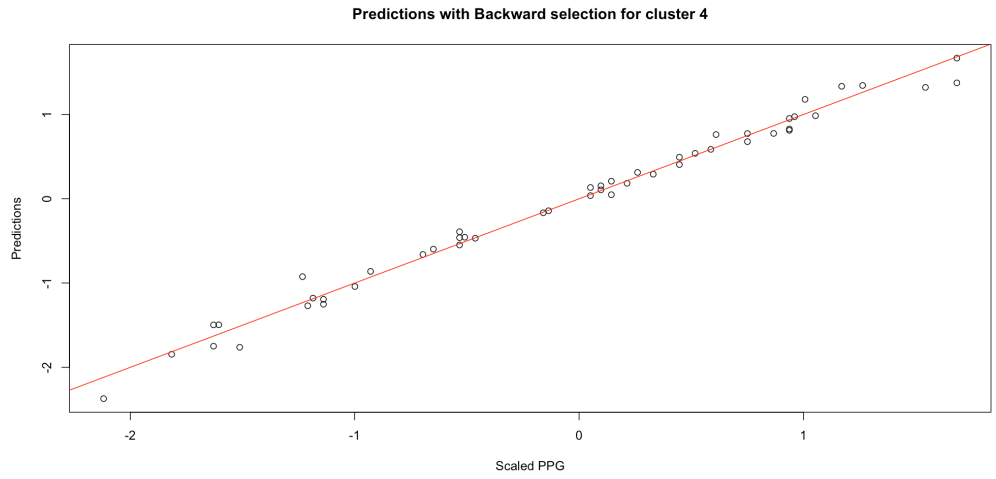


Figure 6.7: Residuals for cluster 4 with linear regression.

As for cluster 4, the superstars one, we can safely say that the results shown in Figure 6.6 are satisfactory. Apart from some forms of outliers, which are shown in the Residual vs Leverage, and a small deviation in the "Normal Quantile-Quantile", Figure 6.6 show a good behavior. The outliers in particular can be explained due to the presence of statistical anomalies in this category, such as James Harden in the season in which he won the League's MVP award. This results are substantiated by Figure 6.7, where it is easy to see that almost all the players stand on or close the bisector. Lots of coefficient influence positevely such a category of players, apart from all the offensive metrics such as shot attempted, assists and rebounds. In particular, USG% sees a particular positive impact, underlining that if there is a player which can be considered a superstar in a team, lots of plays should be directed towards him.

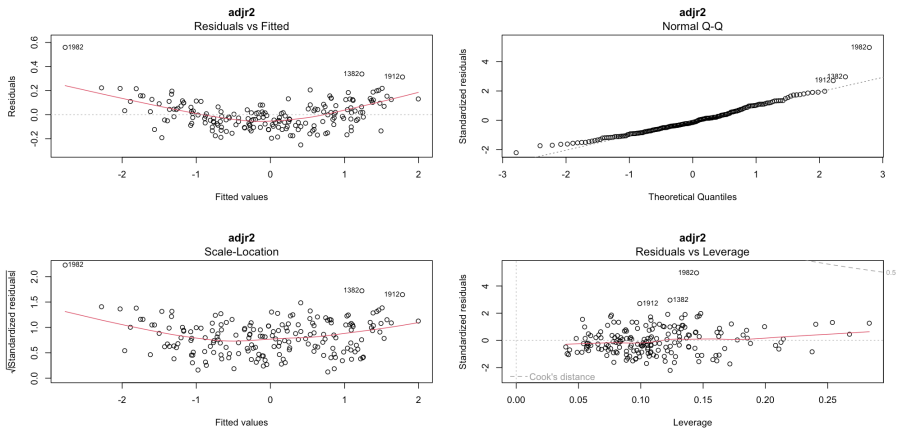


Figure 6.8: Residuals for cluster 5 with linear regression.

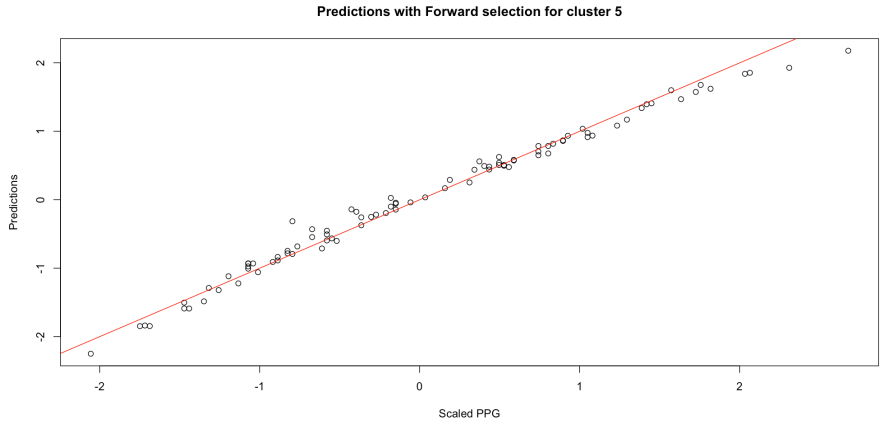


Figure 6.9: Predictions for cluster 5 with linear regression.

Cluster 5 contains players who can be considered pure shooters, contributing to the score of a

team almost only by shooting. As for the residuals, we can see that there are no clear problems, apart from the "Residuals vs Fitted" in Figure 6.8, where the points follow the shape of the red line. What we can than see in Figure 6.9 is solid considering the simplicity of the model we are utilizing. The coefficients which most influence this category of players are, of course, the shot attempts, in particular 2PA and 3PA, which lead naturally to a big coefficient regarding TS%. There is also an interesting accent on the RPG, which positively influence the PPG a shooting player obtain, putting an emphasis on how basketball has achieved throughout the years great fluidity in the roles of the players.

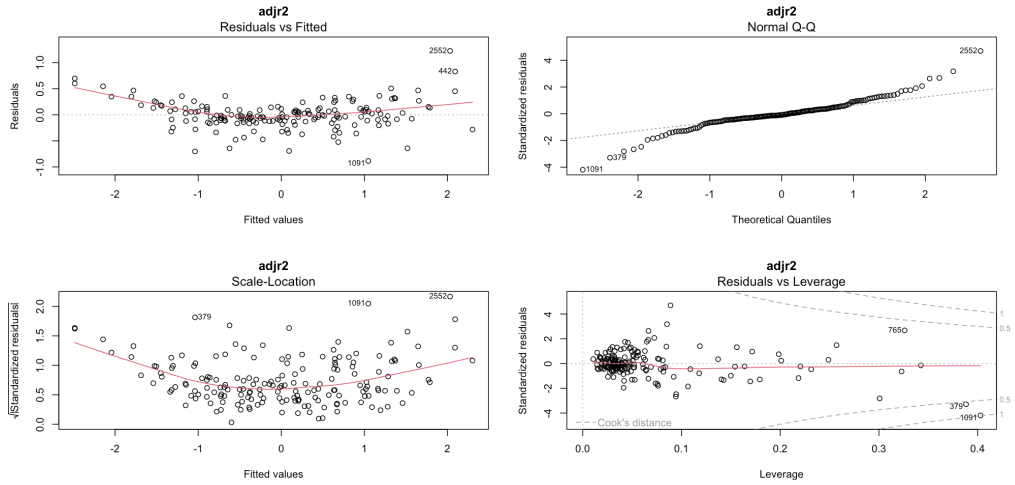


Figure 6.10: Residuals for cluster 6 with linear regression.

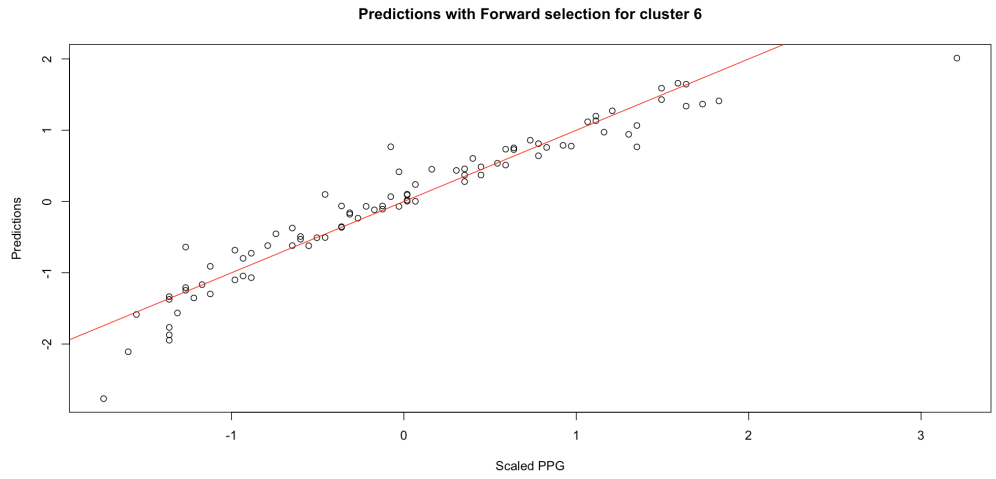


Figure 6.11: Predictions for cluster 6 with linear regression.

We start to see that linear regression models have a similar pattern for our results: the "Residual vs Fitted" in Figure 6.10 shows once again a presence of points around the red line in the plot, while the others look fairly stable, with some occasional exceptions. There are some outliers even at this point, and it can be further elaborated in Figure 6.11, where most points position themselves around the red bisector, with occasional exceptions. In this case, we can see a particular tendency of points to form some more complex behavior, further linear ones, which we can further try to investigate when we move on to more complex techniques. Cluster 6 is populated by mostly big framed players with a tendency for rebounds and not many minutes, and indeed it is underlined that, with more minutes and possesses, they can excell in PPG. The variables which mostly influence this category are TS%, and, as expected, the number of rebounds per game.

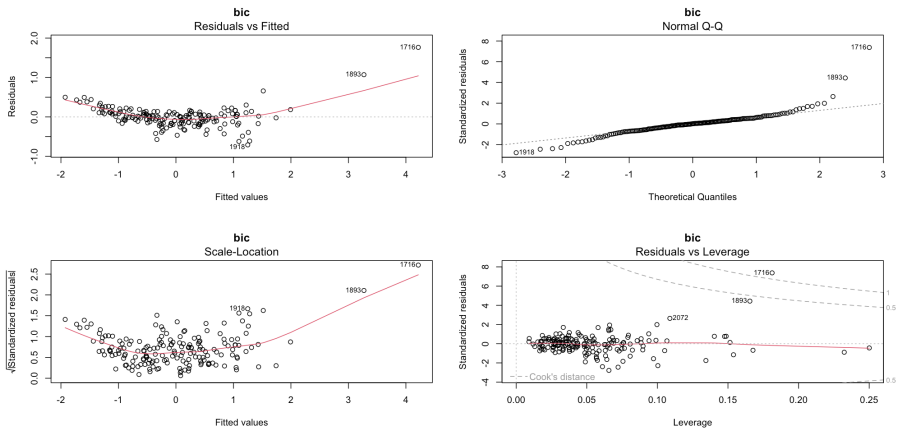


Figure 6.12: Residuals for cluster 7 with linear regression.

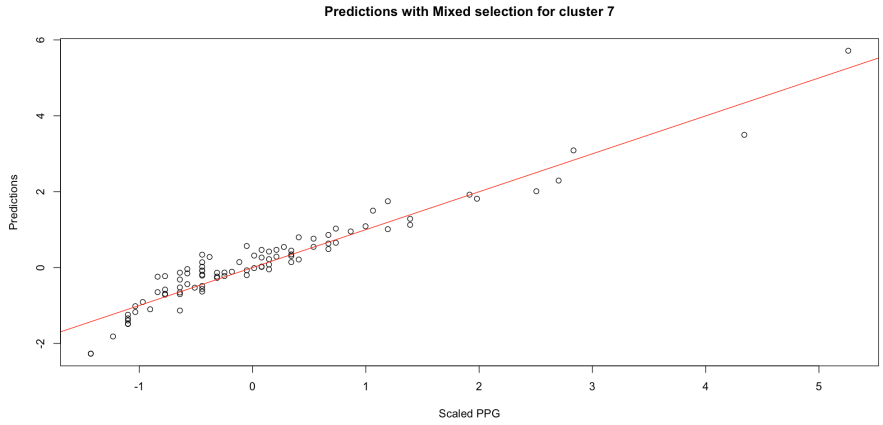


Figure 6.13: Predictions for cluster 7 with linear regression.

As for cluster 7, residuals shown in Figure 6.12 can be deemed very similar to the ones discussed

above. This cluster is actually pretty complex to pin down, since it contains what we earlier defined as "drivers". But even considering its variety of players and preferences, linear regression did a good job at simplifying this problem, obtaining one of the most slim models above the ones obtained by this set of experiments. The interesting coefficients of this model deserve some attention: while it is suggested that a higher USG% can improve the PPG of players in this category, MPG has a negative impact. This implies that, for these players, it is better to have more plays running through them rather than having actually more minutes on the floor. Finally, we want to underline that the predictions in Figure 6.13 on this cluster are acceptable considering the simplicity of a linear model, since most points are in an area close to the bisector.

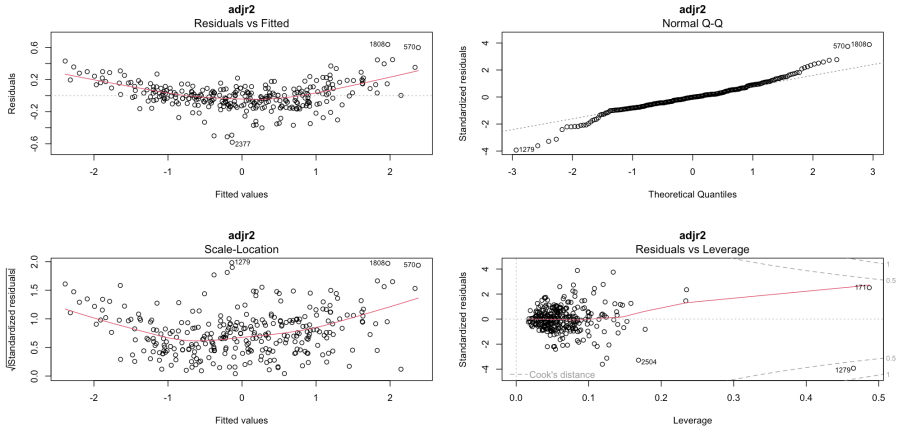


Figure 6.14: Residuals for cluster 8 with linear regression.

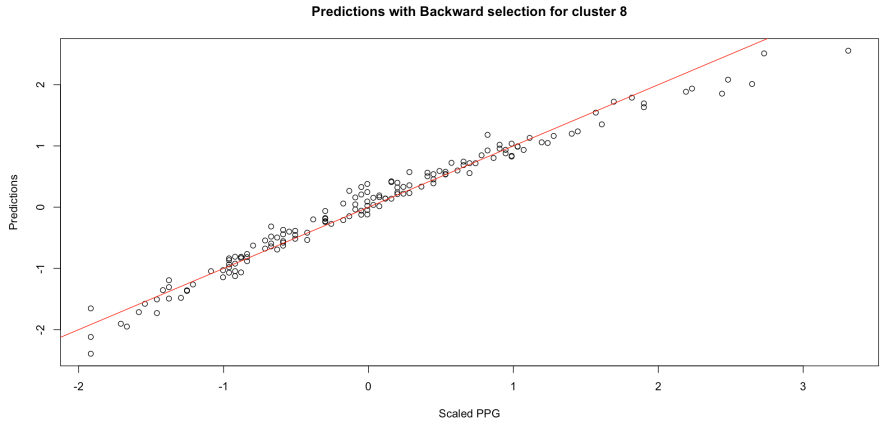


Figure 6.15: Predictions for cluster 8 with linear regression.

The predictions for this cluster present one of the highest Adjusted R^2 above all the other models, which is a particularly interesting feat considering the complexity of this cluster. They are

predominantly players who stay inside the paint and construct their efficiency from rebounds and inside points closer to the rim, which is a wide definition, which allows for a broader interpretation. Even though this is the case, we can see that the residuals from Figure 6.14 follow the similar pattern for all the other models, being that apart from the presence of occasional outliers and the apparent deterministic behavior in the "Residual vs Fitted", these are overall interesting and precise results. Predictions showed in Figure 6.15 also confirm this behavior, since they do not show much dispersion at all. The discovered models contains lots of coefficient actually, but we want in particular to consider a few of them which are interesting. This category of players is required, in order to produce more points, to act in a more modern way, with respect to what someone could expect. Usually, big players are required to stand inside the area and take easy shots towards the rim. Players from cluster 8 are instead more likely to generate points by taking 2 and 3 shots, as well as assists and rebounds.

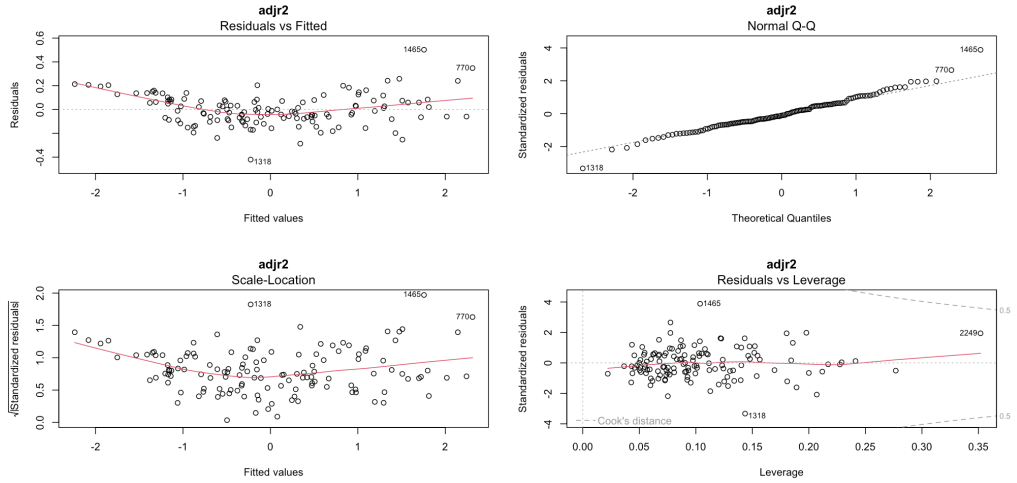


Figure 6.16: Residuals for cluster 9 with linear regression.

Cluster 9 was actually one of the easiest to pin down in our original analysis of categories, being it reserved for the "classical centers". What we see in the results of linear regression is indeed a cluster with good behavior of the residuals, as seen in Figure 6.16, as well as good predictions, with a total absence of outliers. We see that in Figure 6.17 there are not points who are drastically misplaced, and we see an overall uniformity in the predictions, with some bigger variations at the start and at the end of the predictions. The coefficients which influenced the most such a category were actually rather interesting: with respect to cluster 8, there is not a big suggestion in raising the minutes per game of players in this category. And while indeed their USG% is particularly high, since most difficult plays in a tough situation will revolve around the biggest player on the floor, being him the safest way to secure at least two points, there is an accent on the shooting again, as in the previous analysis. 2PA and 3PA are positive coefficients for this model, once again explaining that indeed, there is the need for even bigger players to get more skilled on such aspects of the game, a concept that in the 2010's was still revolutionary, and that just nowadays

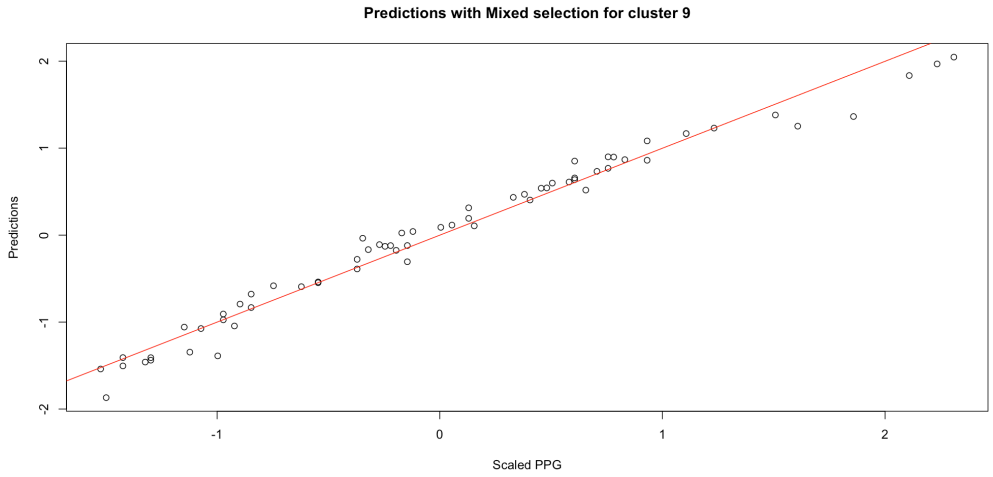


Figure 6.17: Predictions for cluster 9 with linear regression.

is gaining more and more consideration.
The results of this first analysis are, indeed, encouraging for each and every cluster, with exception for cluster 2, which will be excluded from all further analysis for the reasons explained above. We were able to discover 8 robust models for each meaningful cluster. We believe that the recurring problems related to the residuals in the "Residual vs Fitted" is caused by two major causes: first of all, the non-normal behavior of our response variable, which has of course consequences on the goodness of discovered models, and the lack of more data for our analysis. Moreover, the presence of outliers is inherit in such a field of study. Given the inherent volatility of basketball, akin to any sport, instances of statistical anomalies are anticipated year after year, making impossible to create a comprehensive explanations by predictive models. Overall we are satisfied with the numerical results and the graphical ones, considering that this is the most slim and easy to interpret model. We will see how much more complex models can improve these results, but we will need to be careful on a particular information. As the plots for the residuals show, most of the model's covariates were chosen based on the Adjusted R^2 statistic. This implies that, almost at all times, the more complex model was preferred with respect to the simpler one. It brings up the question of how intricate the problem we're studying really is, and aligns with what we're trying to predict, a player's Points Per Game, which is influenced by many different factors. Given this, it makes sense for us to explore slimmer models, that still do a good job of predicting, as they might capture the important factors without adding unnecessary complexity.

6.3 Principal Component Regression

This analysis in particular is interesting in the scope of our study, considering we already applied Principal Component Analysis in Section 4.3 in order to discover clusters in our dataset. We were able that way to find out that with just 4 principal components it is possible to describe almost the entirety of variation expressed by our dataset. Principal component Regression is available in the *R* programming language thanks to the *pls* library, whose functions were used in this set of experiments. The experimental setup can be described as follows.

1. The seed 123 is set at the beginning of this analysis, for repeatability purposes. Then, for each cluster, a principal component regression model is computed, utilizing 20 as a roof for the number of components, *ncomp*. Also, cross-validation is implemented as a further mean to create reliable models. For each one of these, the training indices of each group of players are used.
2. A validation plot is computed, with two metrics, MSEP (mean square error of prediction) and R^2, useful to understand how much variability in the dataset is explained by each component.
3. The number of optimal components is chosen by applying the *onesigma* method, as for Hastie, Tibshirani and Friedman, 2009. It returns the first model where the optimal CV is within one standard error of the absolute optimum. We hence simply use the standard deviation of the cross-validation residuals, in line with the procedure used to calculate the error measure itself. During different tests, it came out that onesigma provides overall more precise models with respect to the randomization approach, as for Van der Voet, 1994.
4. A final *coefplot* is computed for the clusters, with the choice suggested by the *onesigma* approach. The *coefplot* in principal component regression is useful for understanding the contribution of each original predictor variable to the principal components used in the regression. It displays the coefficients associated with each principal component and each original predictor variable. Each coefficient represents how much a unit change in the predictor variable affects the response variable after accounting for the effect of the other variables in the model.
5. Finally, the predictions for the models are plotted, as well as some numerical metrics which are stored in a dataframe, in particular MSE, RMSE and R^2.

Principal Component Regression is anticipated to provide improved outcomes compared to Linear Regression due to its ability to mitigate multicollinearity and enhance model performance. In Linear Regression, when predictor variables are highly correlated, estimation of coefficients becomes unstable, leading to inflated standard errors and potentially misleading interpretations of predictor significance. Additionally, PCR's dimensionality reduction aims at minimizing overfitting, as fewer predictors are utilized in the regression equation. This enhances the model's generalization to new data, leading to improved predictive accuracy. Moreover, PCR tends to be less sensitive to outliers due to the inherent mitigating effect of the principal component transformation, which we have seen can be of major importance in a dataset with outliers such as ours.
We will start explaining the results of this set of experiments by showing the numerical outcomes of the models computed for each cluster. As said, three metrics were computed to give a clear

view of how well each model is behaving. We recall that RMSE stands for "Root Mean Square Error." It is a commonly used metric to measure the accuracy of a model's predictions. RMSE calculates the square root of the average of the squared differences between the predicted values and the observed values, which is just $RMSE = \sqrt{MSE}$.

	MSE	RMSE	R^2
1	0.04725655	0.2173857	0.9445747
2	0.26880427	0.5184634	0.7524244
3	0.13462047	0.3669066	0.8745699
4	0.02109796	0.1452514	0.9789797
5	0.01930319	0.1389359	0.9829199
6	0.06485296	0.2546624	0.9357083
7	0.10021650	0.3165699	0.9229646
8	0.04145602	0.2036075	0.9613650
9	0.03085809	0.1756647	0.9672226

Table 6.2: Performance metrics for PCR models, computed for each cluster.

These results are without doubt interesting. We see models with poorer performances, as well as ones that are similar to the ones we analyzed in Section 6.2. In particular, models related to clusters 2, 3 and 7 have poor performances. And while we can make a case for cluster 2 being underfitted due to the lack of observations, the instances of cluster 3 and 7 are particular and needs to be addressed. Overall, these can be considered, strictly from numerical scores, good enough as models, but they indeed perform worse with respect to linear regression. A first hypothesis on why this is happening can be pretty simple: linear regression may choose, at times, a larger set of variables than principal component regression, which tries to "sum up" the content of the various predictors. We proceed now to show the graphical results for this analysis, starting from the choice in number of components for each model. This analysis in needed in order to understand if there are any analogies between Principal Component Analysis and Principal Component Regression.
At first glance, Figure 6.18 shows us the answer we were looking for: there are no correlations between what we found in Principal Component Analysis and the results from this experiments. The reasons are, of course, many. First of all, the fact that we are trying to predict a single variable instead of "just" summing up the content of the dataset. This implies that the components are required to express the variability of PPG, which is a further requirement that may create the need for a larger number of components. Secondly, we recall that in Section 4.3 we utilized a slimmer version of the dataset, which we did not imply for uniformity in the experiments for this section. Indeed, to a higher number of predictors will naturally correspond a higher number of principal component needed. Finally, we can look at the actual results. Apart from some particular cases like cluster 8, where the number of components needed to reach a justifiable value in both MSE and R^2 is lower, or cluster 2, where those values are never reached, all clusters behave similarly: to reach a value of 0.1 MSE and 0.9 R^2, the number of optimal components *ncomp* is always contained in the following way, $10 \leq ncomp \leq 14$. And while this is a less efficient result than

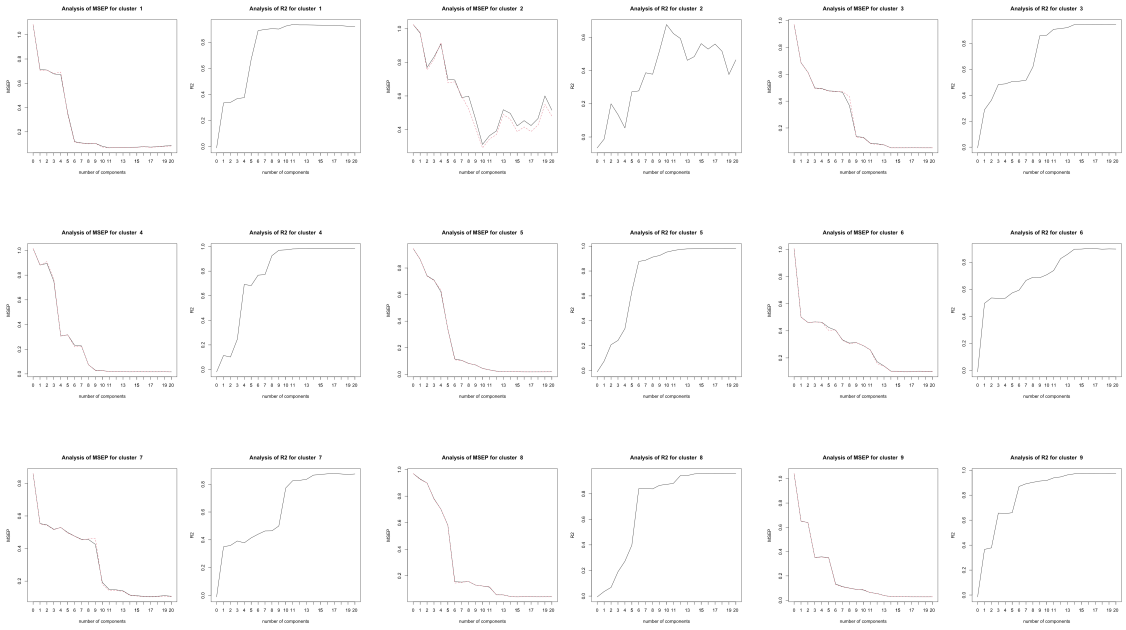


Figure 6.18: MSEP and R^2 analysis for each cluster.

the one obtained in Section 4.3, we can consider this still to be an interesting point of view on our problem. Principal Component Regression is useful to give us a lower bound on the information needed to express the variability of our response variable, PPG. And doing this, not always brings us particularly efficient results, as Table 6.2 shows, meaning that there is a clear limit under which our problem cannot be simplified. Still, we want to take a look at how well a simplified version of our problem works, which can be done by analyzing the predictions for each cluster. Starting from cluster 1, which can be seen in Figure 6.19, we have a robust model with a very low MSE and a solid value in R^2. These influence the predictions of the model significantly, and we are able to see that almost all predictions are near the red bisector, with contained deviations. There is, even in this case, the issue with the usual outlier for this cluster, which not even Principal Component Regression is able to manage. This cluster was able to sum up almost all variability of PPG with the use of just 11 principal components, which is one of the best results among all clusters.
We recall that cluster 2 is not considered in these analysis due to the lack of observations which lead to poor performances in terms of both numerical scores and graphical ones. Just looking at Table 6.2 we are able to see that this model was not able to go further than 0.75 on the R^2. Cluster 3 required 14 principal components, which is the mode among all the models. $\frac{6}{9}$ of the models ended up choosing this number of components to explain the variability of PPG. Cluster 3 in particular is not a very stable model, presenting numerical results which are barely acceptable, and this reflects in the predictions presented in Figure 6.20. Here we see the presence of both observations which do not deviate too much from the bisector, as well as lots of outliers and

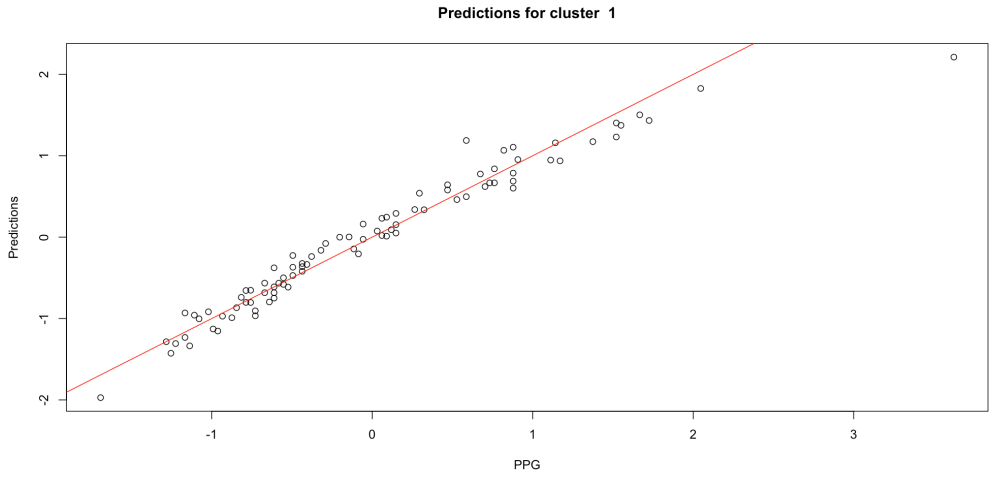


Figure 6.19: Predictions for cluster 1 with PCR.

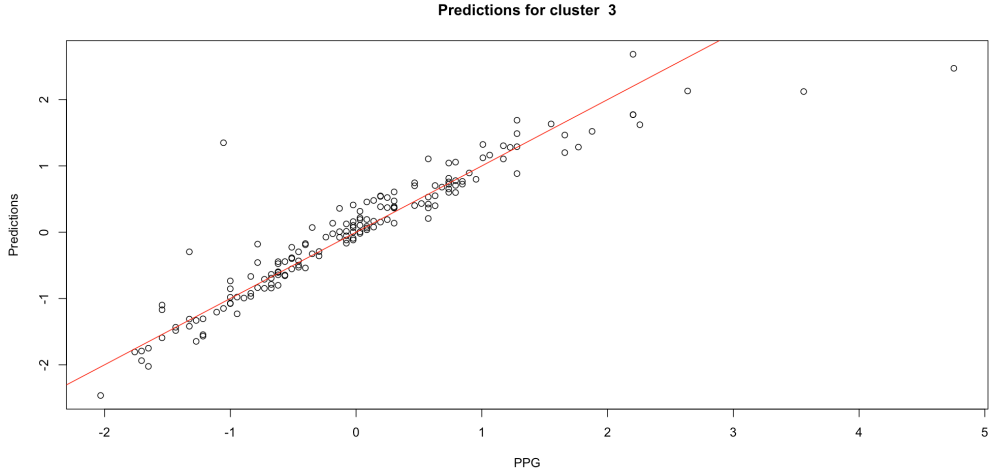


Figure 6.20: Predictions for cluster 3 with PCR.

mispredicted players. And while this is a hard cluster to precisely pin down, containing players who still need development in the league, linear regression with its simplicity was more accurate utilizing 19 predictors. This can be an indicator of the fact that this cluster's best approach to obtain correct predictions may not be a shrinking method. The predictions for cluster 4 are shown in Figure 6.21, and can be considered satisfactory. Indeed this model already presents one of the highest numerical results, implying that we will see next to none variety along the red bisector, obtaining really good predictions. In particular, we want to address that, for this cluster, principal component regression is able to solve one of the issues that linear regression had, which is being robust to outliers. Even in presence of players such as Lebron James or James Harden, this model

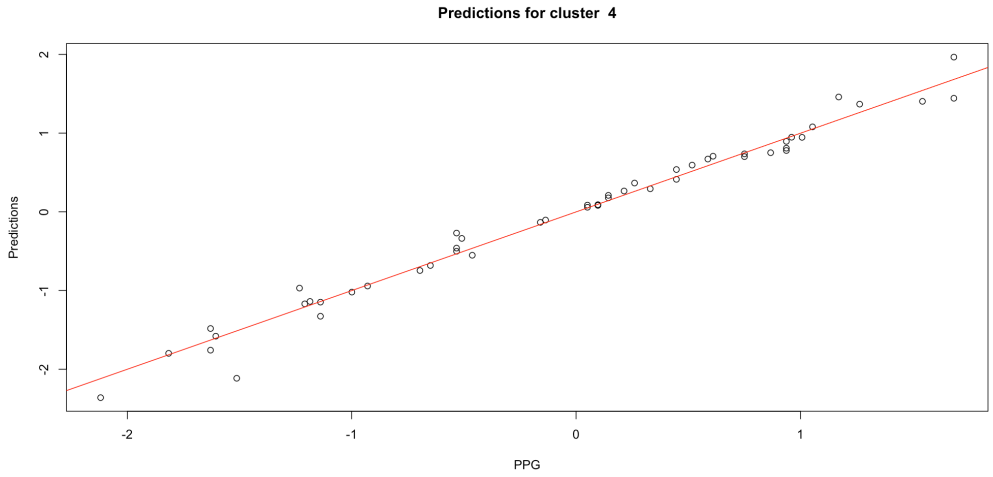


Figure 6.21: Predictions for cluster 4 with PCR.

is able to fit observations pretty consistently. This model, being one of the easiest to address, utilized 11 principal components in order to express PPG's variability.

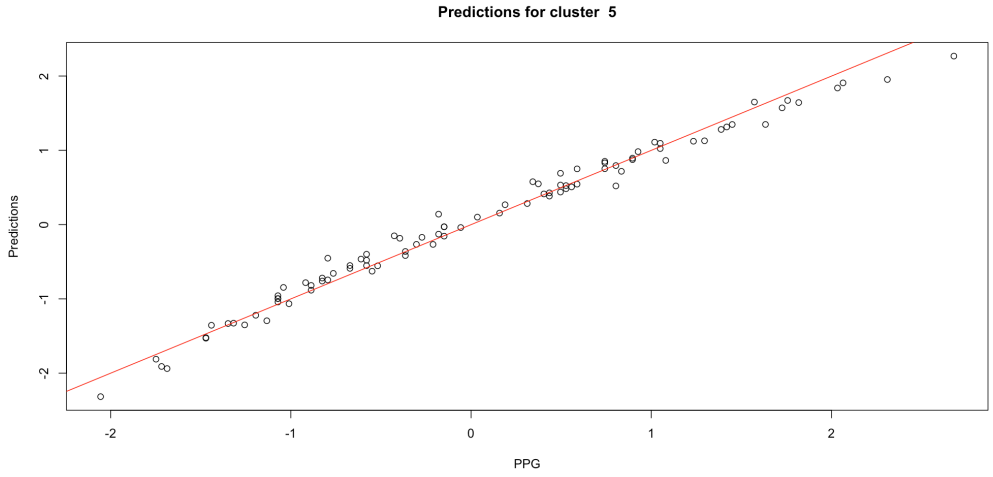


Figure 6.22: Predictions for cluster 5 with PCR.

The model associated to cluster 5 is actually the best one obtained by this analysis, and it shows clearly as well in the predictions. Where we have seen in Figure 6.9 that the model was doing a good job at making predictions, with errors at the beginning and at the end of the curve, this model stays consistent along all the observations. Indeed we can also see an MSE of 0.01, and a R^2 of 0.98, making for a really solid model. Cluster 5 contains the shooters for the league, and being them one of the most searched players among the various NBA team's this model could be

really useful in detecting which athlete has the best chance of developing a high profile in PPG. This model was predicted utilizing 13 principal components.

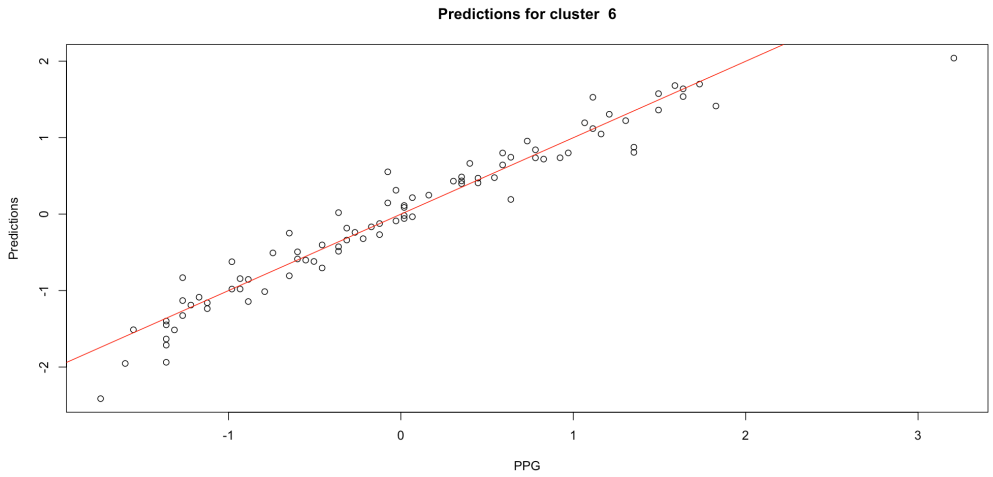


Figure 6.23: Predictions for cluster 6 with PCR.

A couple of considerations can be drawn so far by the few examples of models we saw. As Figure 6.23 shows, the predictions for cluster 6 aren't precise as, for example, the one's from cluster 1, 4 and 5. This is happening, without surprise, for clusters whose categories are harder to precisely define. As we said, cluster 1 contains the best passers, cluster 4 the superstars, and cluster 5 the best shooters, but cluster 3 and 6 both contain players who still need time and development in the league. They are specialized in different aspects of the game, indeed we do not see a model which is mispredicting every observation, but without the minutes on the floor as other players get, it is harder to understand the best way for them to gain points while playing. For this reason, while cluster 6, as well as 3, contains interesting results in terms of predictions, linear regression still does a better job because it utilizes more covariates, trying to explain variability which is not needed for easier to understand clusters. Another hint of this phenomenon is the fact that clusters 3, 6, 7 and 8, the harder ones to define, all utilize 14 components, which is the maximum number these models show.
What we have just said can be easily seen in Figure 6.24 and Figure 6.25, where the predictions are still coherent, but the number of components needed is higher in order to do so. In particular, we can see that principal component regression is once again doing a good job at containing the issues related to outliers. And we can safely say that, in particular for cluster 8, the predictions are at most time around the bisector without much deviations. Indeed, this model still presents a high R^2 and MSE, proving that even in more difficult clusters principal component regression can still do a good job at creating a good model. The reason why we believe this happens only for cluster 8 is related to the presence of more observations in this cluster with respect to others. Of course, having more observations in cluster 3, 6 and 7 would lead to similar models, since the nature of the considered clusters is similar. Moreover, having more observations in the easier to

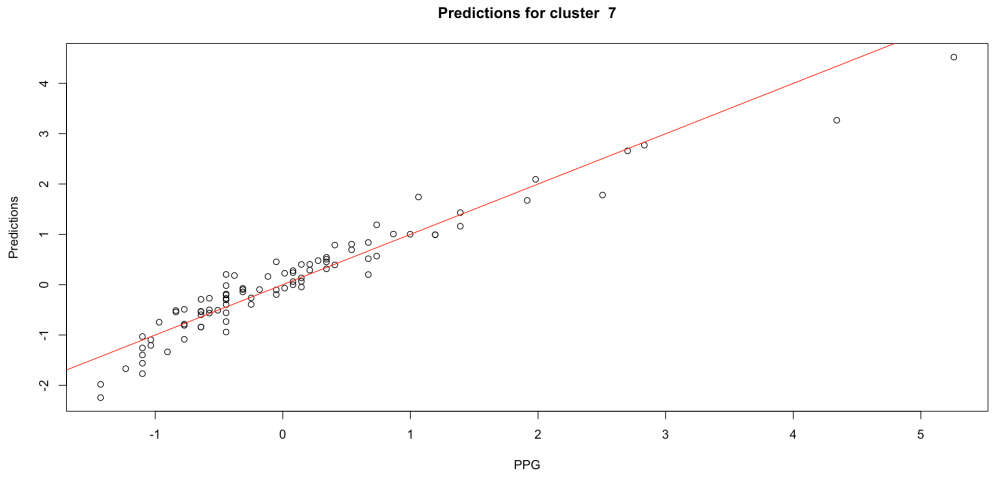


Figure 6.24: Predictions for cluster 7 with PCR.

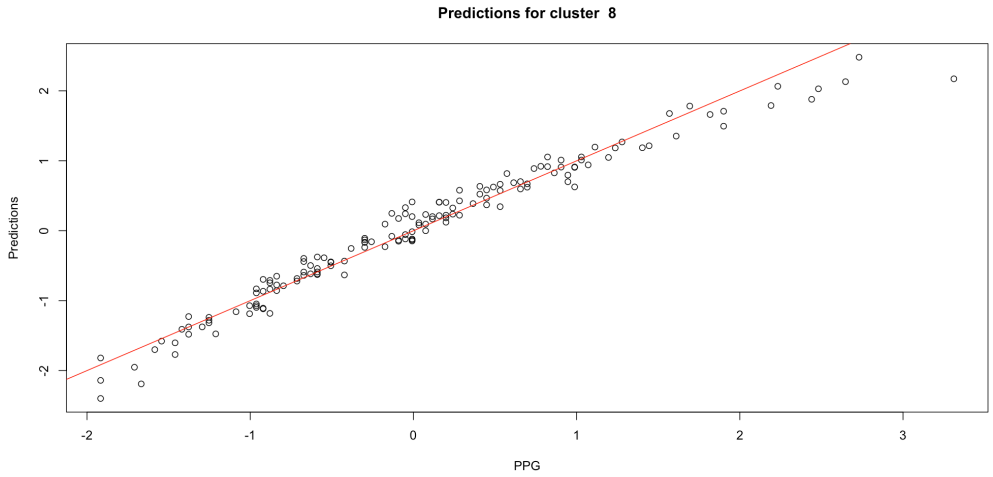


Figure 6.25: Predictions for cluster 8 with PCR.

understand clusters could lead to even more precise models. As already specified, both models for cluster 7 and 8 were created utilizing 14 principal components.

We head to the final model, for cluster 9. It averages the third best numerical results, since it is a fairly precise cluster, containing classic centers. It utilizes 14 principal components still, and the predictions shown in Figure 6.26 are not as good as the ones for cluster 4 and 5, which have similar scores. What we believe is happening here is another proof for the absence of observations in the subset of the dataset. The predictions themselves are rather precise, but we believe a better graphical behavior could be shown if the testing dataset was bigger than the one used for this experiment.

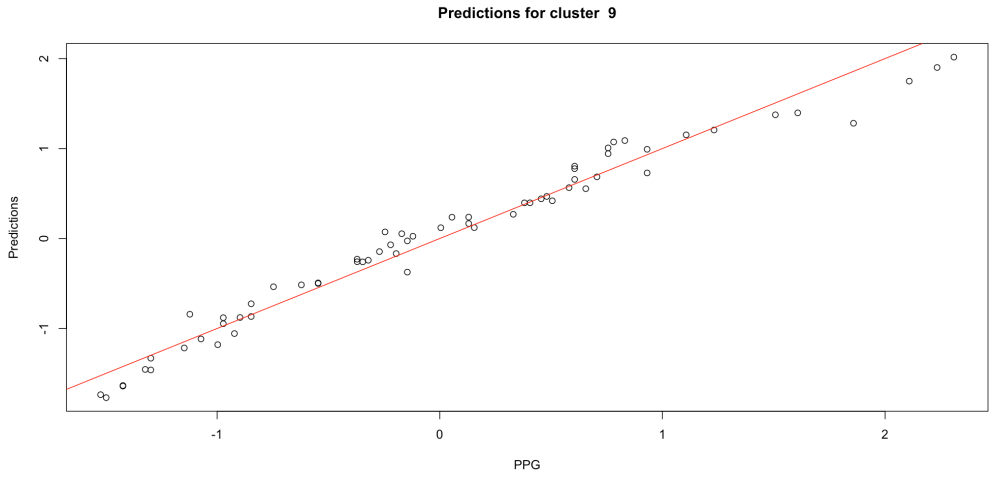


Figure 6.26: Predictions for cluster 9 with PCR.

Overall, we have gained interesting considerations by this analysis, and by its comparison with the results obtained by linear regression. In particular, we have seen that for more "straightforward" clusters, being them 1, 4, 5, and 9, the principal component regression obtains better results with respect to linear regression. A larger number of observations would have lead to even better graphical results, and maybe a lower number of principal components used, although this last claim would need more testing. The same line of reason can be applied for harder to grasp clusters, in particular 3, 6, 7 and 8, since the latter was by far the one with best results due to the presence of more observations in the subset with respect to the other clusters. We conclude that principal component regression is indeed a good way of making predictions about PPG for NBA players, being better than linear regression in the more clear clusters, and worse in the lesser ones, since automatic selection was allowed, in that case, to use a larger number of predictors, in order to make up for the variability not explained in the response variable.

6.4 Random Forest Regression

In the realm of predictive modeling, it is essential to explore diverse regression techniques to comprehend their effectiveness in addressing complex and multi-dimensional datasets. We analyzed and utilized so far Linear Regression and Principal component Regression, and their results can be categorized as satisfactory. Linear Regression, a fundamental approach, seeks to establish linear relationships between predictors and a response variable, making assumptions of homoscedasticity and independence of residuals. PCR, on the other hand, leverages dimensionality reduction via Principal Component Analysis, aiming to alleviate multicollinearity concerns and improve prediction accuracy, but still applying a least squares approach to the obtained components. However, these techniques may encounter limitations when faced with intricate relationships or non-linear patterns within the data. Random Forest Regression, an ensemble learning algorithm, offers a promising solution by aggregating multiple decision trees and introducing non-linearity, enabling the capture of intricate interactions among variables, as explained in Section 5.4. In this context, the exploration of Random Forest Regression as a complementary method stands to unveil its potential in addressing the shortcomings of Linear Regression and Principal Component Regression, while yielding insights into its applicability and performance on our case study.
The *randomForest* package in the *R* programming language allows us to conduct our experiments with this technique, and, once again, we begin our treatment by explaining the setup that has been used for the analysis of each of the nine clusters.

1. For each cluster, a random forest model is built in the *R* programming language, using the call to function *randomForest* with the following parameters. The response variable, PPG, the training indices for each cluster, and three training parameters. Firstly, the number of variables randomly sampled as candidates at each split, or *mtry*. Given p the number of predictors, the default values are different for classification problems, where $\sqrt{p}$ is used, and regression, where $\frac{p}{3}$ is used. Hence, the number of covariates is divided by 3 and rounded without any decimals. Secondly, *ntree* is the training parameter which specifies how many trees are builded during this phase. A common practice is to set the *ntree* parameter to a larger value, such as 500 or 1000, and then evaluate the model's performance. If the performance metrics stabilize or show minimal improvement after a certain number of trees, it is possible to stop increasing the number of trees. In our case, 500 was a good tradeoff between accuracy and computational efficiency. Finally, the *replace* parameter was set to TRUE, to enable bootstrapping with replacement.
2. The numerical scores, MSE, RMSE and R^2 are computed and inserted in a dataframe in order to be analyzed, and the predictions are plotted to validate the model.

These are all the steps that we utilized for the random forest regression method. Note that, differently from the linear regression and principal component regression analysis, some interpretability information is lost in order to get a more efficient model. Explained our experimental setup, we will now provide a table with the numerical scores of the random forest models for each cluster. Table 6.3 shows clearly that this experimentation did not go as well as the previous ones. Recalling the boundaries we used for Principal Component Regression, the MSE of this model never goes below 0.1, except for model 9, and the same happens for the R^2, where the threshold of

	MSE	RMSE	R^2
1	0.14104298	0.3755569	0.8345763
2	0.27891650	0.5281255	0.7431108
3	0.18734953	0.4328389	0.8254406
4	0.11074226	0.3327796	0.8896655
5	0.16042099	0.4005259	0.8580546
6	0.22560899	0.4749831	0.7763435
7	0.30601510	0.5531863	0.7647692
8	0.18518841	0.4303352	0.8274133
9	0.08772464	0.2961835	0.9068191

Table 6.3: Performance metrics for Random Forest models, computed for each cluster.

0.9 is never reached except for that one occasion. This is a clear issue. We are dealing with a model that cannot be considered at all stable or utilized, and we have to investigate the reasons why this might be happening. First and foremost, one of the main reasons for suboptimal results could be the presence of noisy or irrelevant features in the dataset. Since Random Forest constructs trees based on random subsets of features, irrelevant or noisy variables can lead to the creation of trees with weak predictive capabilities, ultimately affecting the overall ensemble performance. On this topic, a dataset not generic as the one we are utilizing, and more specific on offensive capabilities, could lead to a more significant result. In particular because Random Forest Regression is the only model so far who is not able to make a selection of which are the most meaningful variables. Moreover, Random Forest can struggle when dealing with imbalanced datasets, where one class significantly outweighs the others. This can lead the model to prioritize the majority class and neglect the minority class, resulting in biased predictions. This should not be the case for our dataset, since we have seen that at least a few coefficients are always needed in order to get a full comprehension of the variability in the dataset. The only help we can get our model is to use the slimmer version of our dataset we already considered for Section 4.3, which removed all the information which were not strictly related to offensive parameters, which the models we analyzed so far were able to mitigate. The results for this analysis can be found in Table 6.4 and are even poorer, which bring us to make us further considerations on why the random forest regression is failing for our dataset. Further experimentation with random forest would of course need a new dataset. In the realm we are trying to analyze it is clear that this technique cannot create a complex model with just these information. What could indeed help is a larger set of offensive metrics. We believe that the second example, created via the use of a slimmer dataset, was following the right track, but the lack of observations and of more meaningful covariates ended up creating a model which still fell too short. In a low-observation scenario, the randomness introduced by the model's feature sampling during the construction of individual trees may lead to high variance and unreliable predictions. Since each tree in the ensemble is constructed using different subsets of the data, the aggregated predictions might not adequately capture the underlying patterns or relationships within the data. Additionally, the model might struggle to identify meaningful splits

	MSE	RMSE	R^2
1	0.3135424	0.5599486	0.6322586
2	0.5996733	0.7743857	0.4476856
3	0.3123129	0.5588496	0.7090084
4	0.3548560	0.5956979	0.6464508
5	0.4033897	0.6351297	0.6430685
6	0.3581553	0.5984608	0.6449443
7	0.4317126	0.6570484	0.6681468
8	0.4383922	0.6621119	0.5914395
9	0.2890561	0.5376394	0.6929653

Table 6.4: Performance metrics for Random Forest models, computed for each cluster on a smaller set of predictors.

and decision boundaries in the data due to the limited instances available for analysis.
All the aforementioned reasons bring us to believe that the actual predictions are not fundamental to show, due to their inability to capture any meaningful information in our dataset. Even in the special case of the model for cluster 9 which utilized all the covariates, where the R^2 was higher than 0.9 and the MSE smaller than 0.1, the predictions ended up being not much meaningful, probably due to an overfitting of the training data.

6.5 Shrinkage Methods

In the realm of sport's predictive modeling, where the task is to unveil intricate patterns within data and craft accurate predictions, traditional approaches like linear regression and principal component regression often encounter challenges posed by multicollinearity, noise, and model complexity, as we were able to see so far. Shrinkage methods, such as Lasso and Ridge regression, offer a promising avenue to transcend these limitations. These techniques, rooted in the broader context of regularization, extend the principles of linear regression by adding penalty terms that constrain the coefficients of predictors. Lasso employs L1 regularization to induce sparsity, driving some coefficients to exactly zero, thus facilitating feature selection and potentially yielding simpler and more interpretable models. Conversely, Ridge regression utilizes L2 regularization to control the influence of coefficients, which can mitigate the impact of multicollinearity and reduce the vulnerability to noise. In this series of experiments, we delve into the implementation and evaluation of Lasso and Ridge regression, scrutinizing their ability to counteract overfitting, enhance predictive accuracy, and offer insights into feature importance. Through these investigations, we aim to not only advance our understanding of shrinkage methods but also to harness their potential in overcoming the limitations inherent in linear regression and principal component regression for the problem at hand, with the hope to achieve a more robust and reliable predictive model.
The functions needed for this specific analysis in the *R* programming language are present in the *glmnet* package.

6.5.1 Ridge Regression

We logically start from the type of model which not shrink the coefficients exactly to zero, and talk hence about the experimental setup for Ridge Regression. For this analysis, as well as any other, the seed was set to 123 for purposes of repeatability. We can describe the setup as follows.

1. For each cluster, create a ridge regression model without cross validation, and one with this further requirement. The models are created starting from the training indices of the original datasets.

2. Two plots are computed. The model without cross validation is used to plot the behavior of coefficients with an increasing value of $log(\lambda)$, while the cross validation one is utilized to compute the best possible λ. The two dashed lines are the values of the minimum λ and the minimum one with one degree of freedom, which is usually more penalizing.

3. A new model is computed for each cluster, and it is trained using the minimum λ obtained at step [2].

4. The numerical scores for the optimal model are computed, in particular we save the correspondent λ, the MSE, as well as the explained deviance and the R^2.

5. The predictions for the optimal model are plotted utilizing the testing indices, together with two graphical representations of the final penalization chosen.

	MSE	ED	R^2
1	0.06768770	0.9582596	0.9375058
2	0.25030115	0.9558004	0.8111838
3	0.05395081	0.9541981	0.8798625
4	0.02179382	0.9862570	0.9831053
5	0.02063096	0.9820917	0.9770505
6	0.11170450	0.9240984	0.9420257
7	0.11151018	0.9155337	0.9336214
8	0.04308615	0.9670073	0.9562272
9	0.02687474	0.9788065	0.9662002

Table 6.5: Performance metrics for the Ridge Regression models, computed for each cluster.

The numerical results for these models can be seen in Table 6.5. These are pretty in line with what we were able to identify with the previous models. We see a different parameter with respect to the RMSE, being the ED, or explained deviance, of each model. These models are behaving similarly, for example, to the ones we already seen for linear regression and PCR. In the particular case of cluster 2, there is a MSE of 0.25 which makes it its analysis further more easy to discard. Overall, we can be satisfied with these models, even though the penalization factor, as we will see, never gets too high, implying that in order to obtain this score in terms of explained deviance almost all variables are needed. We can look cluster by cluster at the graphical predictions for this model. Together with the predictions for the different models, we will provide two graphical examinations, containing the trend of the penalization factor with an increasing value of $log(\lambda)$ as well as a plot showing the explained deviance with the current choice of λ.

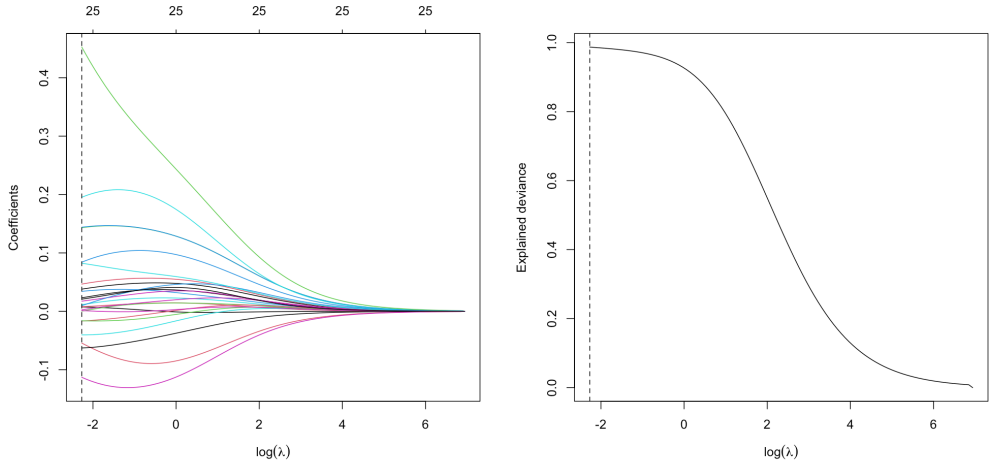


Figure 6.27: λ choice for cluster 1 with Ridge Regression.

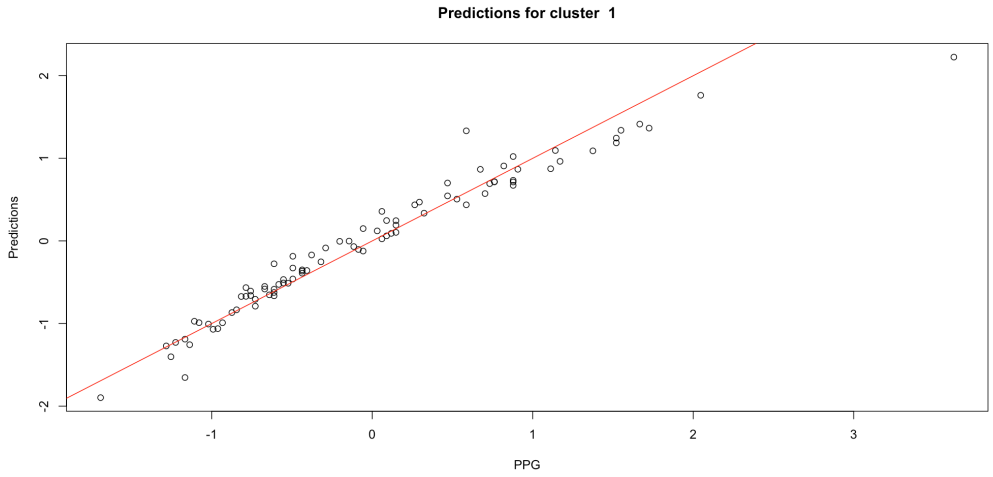


Figure 6.28: Predictions for cluster 1 with Ridge Regression.

Figure 6.27 shows practically what we were just talking about. The penalization factor is indeed very low, but this comes with the obvious consequence of a better explanation of the response variable as a whole. In Figure 6.28 we are able to see that indeed the points are almost at all times close to the bisector of the plot. As in Section 6.2, we are able to see that, for cluster 1, the coefficients which prominently influence the PPG for a player are APG and 2PA. Also 3PA, in this case, have an importance on the final outcome. Most of the other coefficients are shrinked, to the point of having a really small impact on the final outcome.

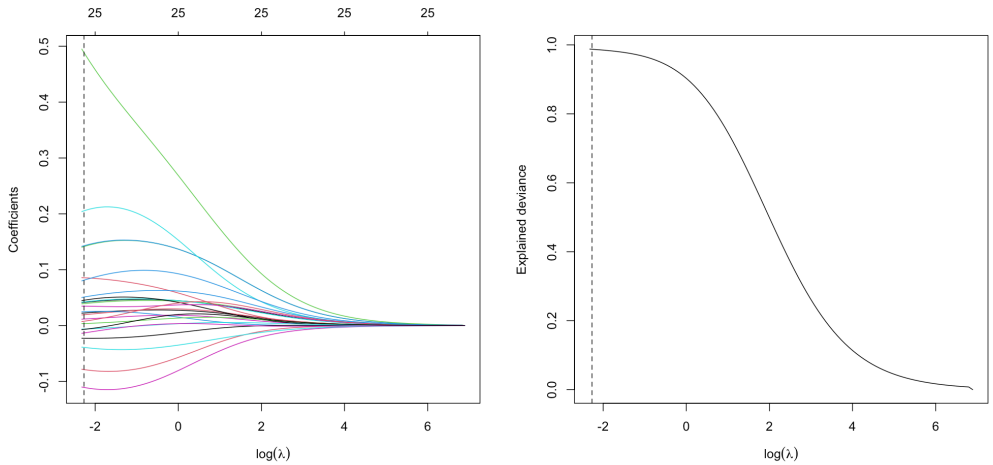


Figure 6.29: λ choice for cluster 3 with Ridge Regression.

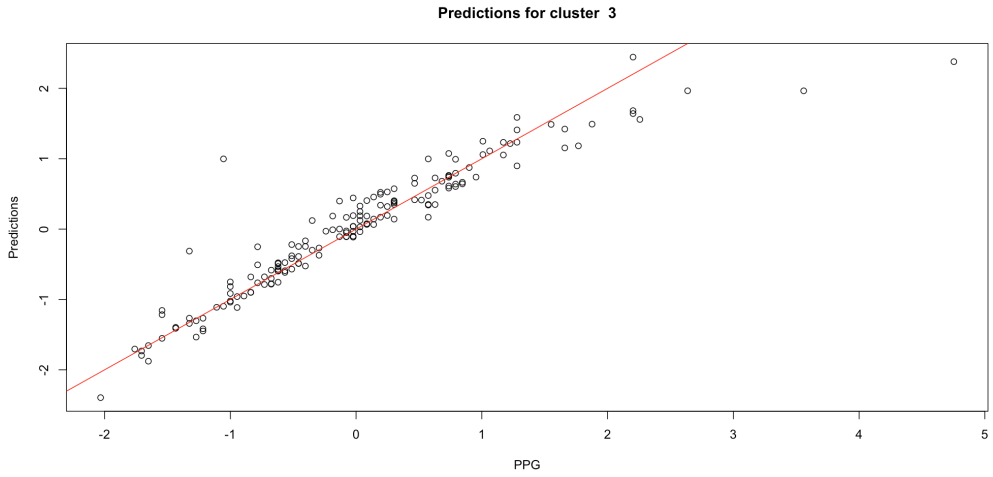


Figure 6.30: Predictions for cluster 3 with Ridge Regression.

With respect to cluster 1 results, which related to a pretty stable and reliable model overall, what we see in cluster 3 could be anticipated slightly from Table 6.5. We have one of the smallest R^2 values, which can indeed be linked to the fact that this is one of the harder to grasp clusters. And while the penalization factor is again low, the predictions this time around do not behave as well. Most points are indeed in a contour of the optimal results, but we fail to see a reliable predictor, considering also that this cluster is filled with outliers that we failed at almost all times to fit well. The results from Figure 6.30 are indeed in line with the models obtained so far.

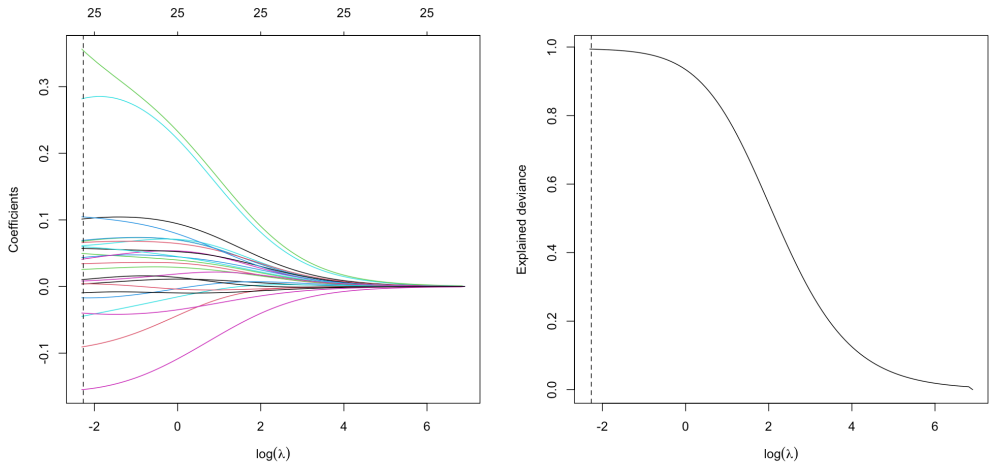


Figure 6.31: λ choice for cluster 4 with Ridge Regression.

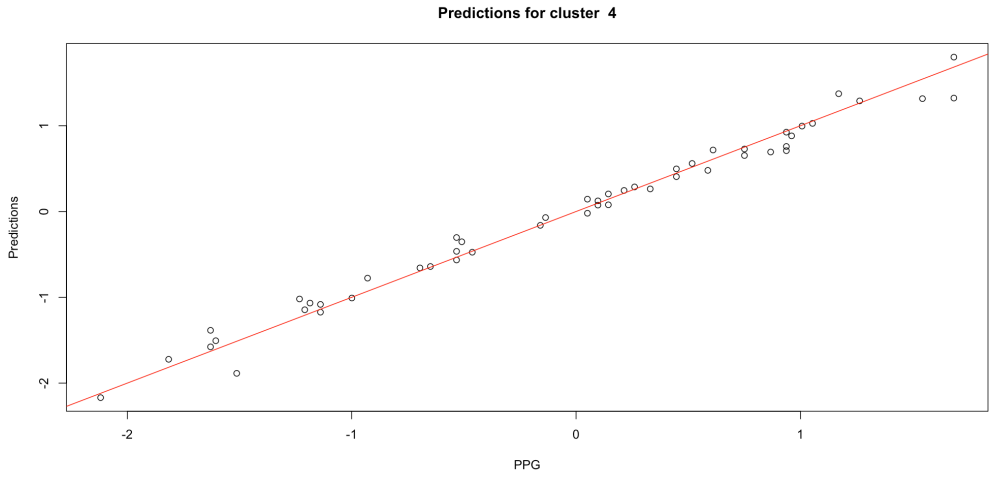


Figure 6.32: Predictions for cluster 4 with Ridge Regression.

Cluster 4 was in all the analysis seen so far, and still is, one of the easiest clusters to analyze. Indeed, superstars has so many ways of efficiently increase their PPG, that pretty much any predictor was able to obtain a solid result, as the one we are able to see in Figure 6.32. As all the other cases, the penalization factor is low, and the most influential coefficient is the USG%. This does not come as a surprise, since any team with a superstar want the ball to be in their hands during the most plays, being him the best player at generating points in a variety of ways.

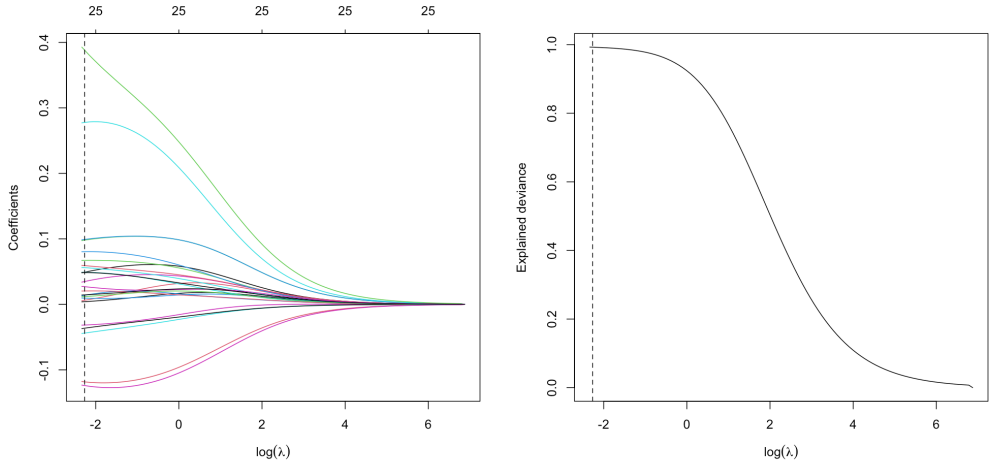


Figure 6.33: λ choice for cluster 5 with Ridge Regression.

Cluster 5, being it populated by the most efficient shooters in the league, is once again one of simple analysis, and the ridge regression model does not fail to gain results. We are able to see a

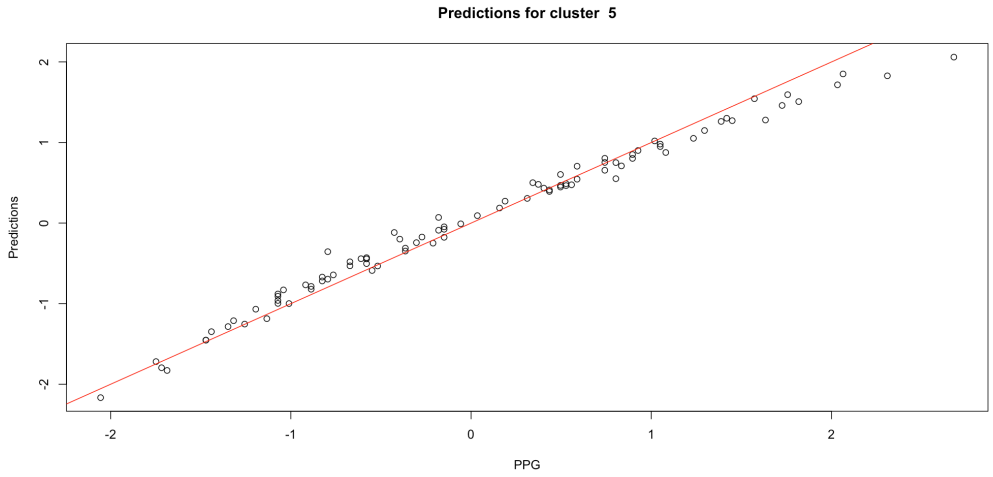


Figure 6.34: Predictions for cluster 5 with Ridge Regression.

particularly interesting result, with a very low presence of outliers, and most points gathering in the near proximity of the bisector, as in Figure 6.34. Again, players who are really specialized in shooting are one of the most important assets in the modern NBA: for this sole reason, cluster 5 is the only other case, other than 4, in which USG% has such an high impact on the overall PPG. Without surprise, the second most important coefficient is, in this case, the TS%.

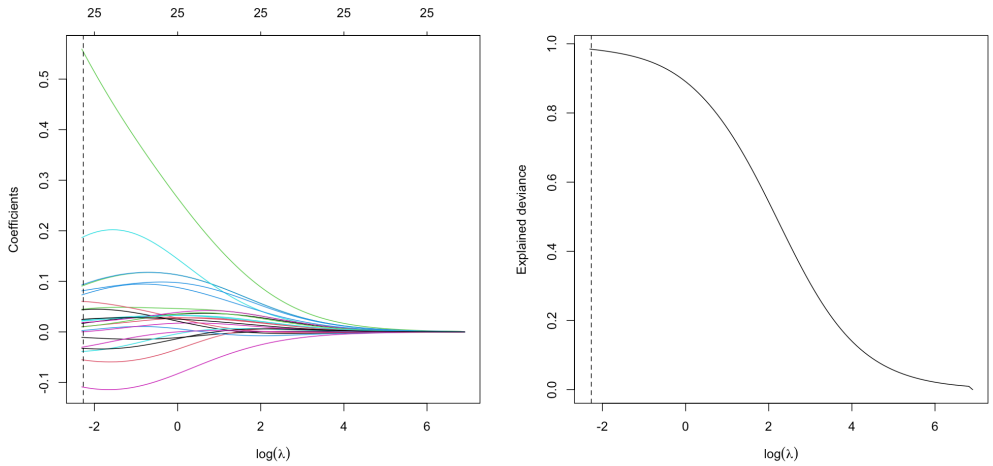


Figure 6.35: λ choice for cluster 6 with Ridge Regression.

We can see from this set of results another situation which is in line with the previously analyzed models. Cluster 6 is one of the harder to grasp group of players, meaning that achieving on it a R^2 of 0.94 with an explained deviance of 0.92 is a good result, but not one of the most stable. The

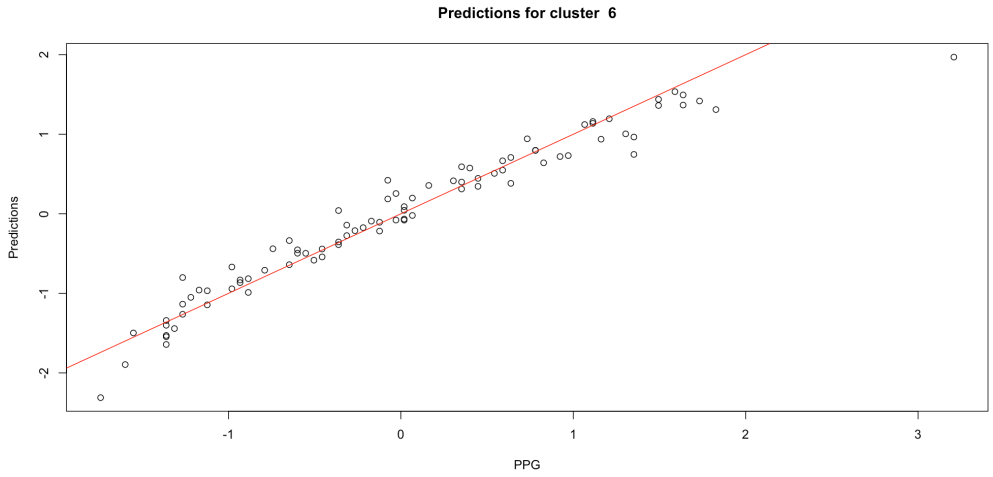


Figure 6.36: Predictions for cluster 6 with Ridge Regression.

predictions for this group reflect this behavior, having most points indeed in the proximity of the bisector, but with smaller or larger deviations. Although the predictions aren't as precise as they can be, we can underline that still the predictions are in line with the results from Figure 4.2.1, since this model is suggesting that cluster 6 players should work on their rebounding proficiency to gather more points, as it is underlined by the positive coefficient in RPG.

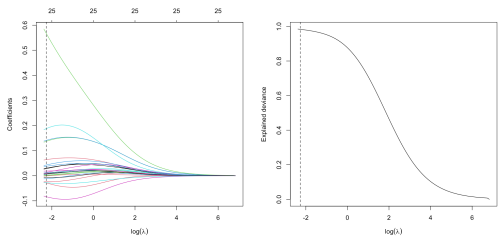


Figure 6.37: λ choice for cluster 7 with Ridge Regression.

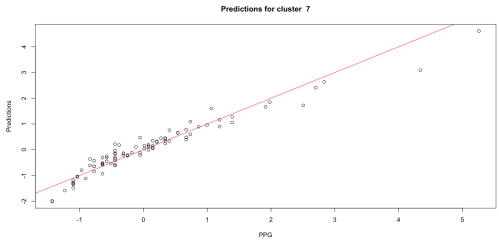


Figure 6.38: Predictions for cluster 7 with Ridge Regression.

Being cluster 7 and 8 so similar to what we are able to see in Figure 6.36, we gathered the results. Again, the penalization factor from Figure 6.37 is low, creating a stable but somewhat reliable model, since it is able to satisfy the lower bound requirements on a numerical score stand-point. The R^2 is indeed higher than 0.90 and the MSE is just high the 0.10 bound. We believe once again, as explained in Section 6.3, that the main reason why cluster 7 and 8, being similar semantically as they are, create different models in term of efficiency is related to the difference in the number of observations. Cluster 8 sees a much higher number of observations with respect to cluster 7, and this create not only smoother predictions, as it can be seen in Figure 6.40, but better scores as well. The MSE drops to 0.04, and the explained deviance goes up by 5 units with respect to the

previous cluster. With all of this being said, we can conclude once again that a higher presence of observations would lead to a much more uniform and reliable set of predictors overall.

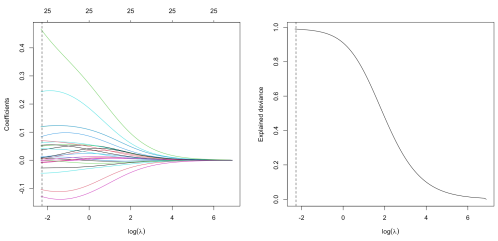


Figure 6.39: λ choice for cluster 8 with Ridge Regression.

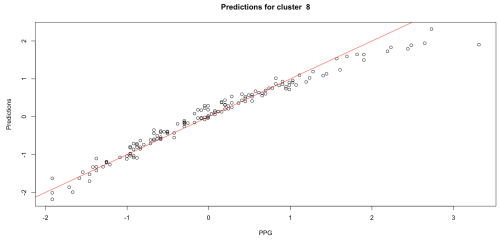


Figure 6.40: Predictions for cluster 8 with Ridge Regression.

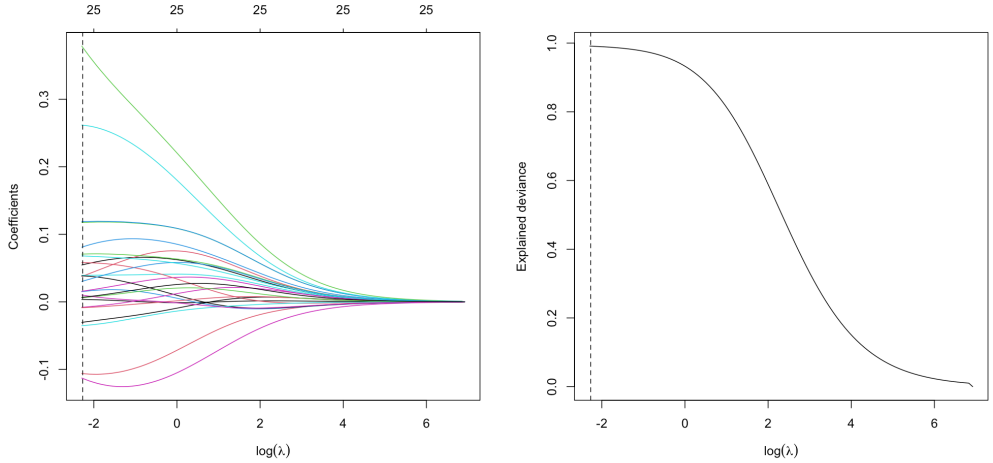


Figure 6.41: λ choice for cluster 9 with Ridge Regression.

We finally conclude with cluster 9, one of the easier to ones to grasp, which indeed reflects this behavior with very promising results. The nature of the predictions, Figure 6.42, which may seem at first glance scattered around the red bisector, is merely an effect of the low presence of observations in the testing subset. This model is instead one of the best among the ones discovered, and indeed confirms our claims on the nature of the cluster itself. An accent is put in this model on the RPG, as well as in the shooting. Both 2 and 3 point shoots predictors have positive coefficients, where the first have a higher impact than the second. This is an interesting result with respect to what we assumed in Chapter 2, which is the rising attention in the role of modern centers.
In conclusion, the outcomes obtained from the ridge regression analysis have yielded results that align incredibly well with those obtained through both linear regression and principal component regression methodologies. This consistency across diverse modeling techniques underscores the robustness of our findings. Moreover, it is noteworthy that the computed penalization factors for

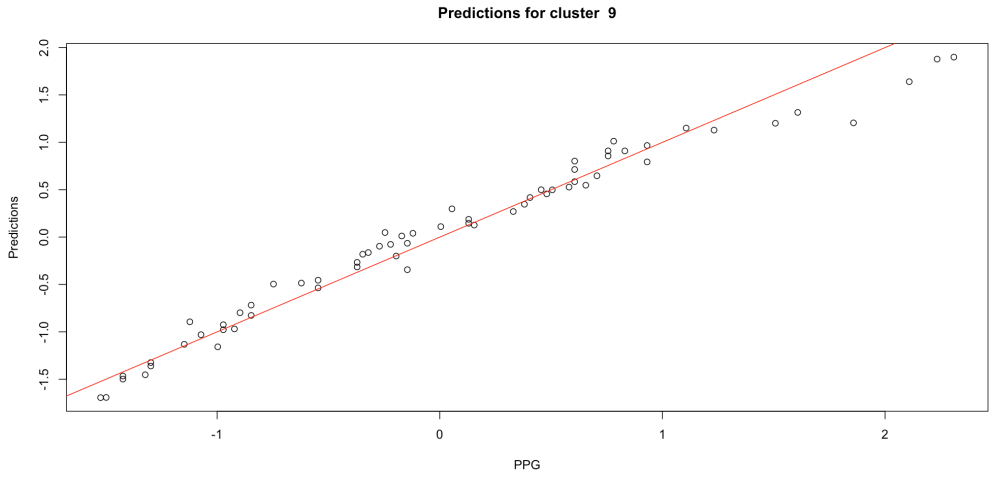


Figure 6.42: Predictions for cluster 9 with Ridge Regression.

all the clusters found in the original dataset were consistently low. This observation highlights the relatively gentle regularization impact of ridge regression on the model coefficients. The nature of these results not only reinforces the validity of our approach but also suggests that ridge regression has facilitated the development of predictive models that effectively manage multicollinearity without introducing excessive penalty to the coefficients. Furthermore, as we reflect throughout the current section, it becomes evident that with larger sample sizes and more advanced predictor variables, the potential for further enhancing model performance is promising. This can be seen in the results obtained for cluster 8, as well as in the fact that reliable results were obtained just considering standard performance metrics. Having access to advanced offensive measurements would of course yield more deep and useful considerations. Therefore, our work stands as a foundation in exploring the potential of ridge regression, as well as the other methodologies we analyzed, in complex predictive modeling scenarios. As we move forward, we believe that having access to more advanced datasets will solidify the efficacy of the techniques we used. In this context, our study serves as a proof of concept for more intricate investigations, showing that the present findings are just the top of the iceberg of what could become state of the art methodologies for both coaches and general managers in the NBA.

6.5.2 Lasso

In attempting to extend our analysis to include Lasso regression, an unforeseen challenge arose that made impossible its applicability to our dataset. Upon implementation, the warning message from listing 6 surfaced. It indicates that the Lasso regression procedure resulted in a list of models with extremely few nonzero coefficients, rendering the visualization and interpretation of the regularization results ineffective. Not only this, but the results of all numerical scores as well were,

Listing 6 Warning obtained for every cluster while applying Lasso Regression.

```
Warning: 1 or less nonzero coefficients;
glmnet plot is not meaningful.
```

without surprise, set to 0, meaning that not a single significant model was produced. This outcome can be attributed to the relationship between the L1 regularization term employed by Lasso and the specific characteristics of our dataset. As Lasso aggressively shrinks coefficients toward zero and performs feature selection by pushing some coefficients exactly to zero, the resulting model may encounter difficulty accommodating the complexity of the relationships among predictors and the response variable. Ridge regression solved this issue by applying a gentle penalization factor, which is not as easy in Lasso regression. This phenomenon can be particularly pronounced when dealing with datasets of limited sample size or intricate predictor relationships. Where the first issue has already been explained, the second one is once again related to the nature of the used predictors. The ones we considered relate to the main metrics utilized to measure players in the basketball realm. A use of advanced metrics may favor the shrinking of coefficients towards zero: indeed, the three point shot for a solely post up player may as well be 0, leaving more floor to shrink most of the coefficients. Moving forward, a more nuanced approach to dataset preparation, feature selection, or modification of the regularization parameter may be necessary to overcome this challenge and unlock the potential benefits of Lasso regression in the analysis of NBA players.

7 Conclusions

We have dissected, throughout this thesis, the possibility of utilizing data mining and machine learning techniques to the real of sports analysis, with the specific focus of NBA basketball. The main achievements we wanted to obtain were two:

1. Obtain Machine Learning models able to differentiate players in ”categories”, which is a more complex definition of the traditional ”position” concept in basketball.
2. Find the best Data Mining technique that, for the categories we found, is able to predict the best way for players to impact the offensive outcome of a game.

We can conclude that, in general, these objectives were satisfied. Even with a dataset in which advanced offensive statistics were not recorded, we were able to tackle the problem using the traditional offensive ones, such as points, assists, rebounds and shooting percentages.
Chapter 4 showed that the best, most solid model for categorizing the starting dataset is the **k-means algorithm**, with a preference for Hartigan-Wong implementation, even though its comparison with other similar algorithms proved that differences are minimal. We were able to discover 9 categories, some more precise and coherent with the initial dataset evaluations, some more vague, and this can be attributed to the presence, in those cases, of players with less minutes played, which creates an unbalanced input. A particularly interesting result was re-applying the same algorithm inside one of the more defined categories, which was done in Subsection 4.2.1.4, referring to the NBA’s ”superstars”. In that case, the analysis showed much more precise predictions, which is a nice future work that can be done for this study: find out how much a recursive application of this algorithm in the discovered clusters would help boost precision and obtain more complex definitions, with the possible downside of overfitting the dataset. In Chapter 4, we were also able to discover a method which was not a good fit for our problem, being it DBSCAN, which helped show the complexity of the problem addressed. Finally, an experimentation with Principal

Component Analysis was carried, and was able to discover four principal components to explain 91.34% of the variance in our dataset. With this result, combining all the principal components with one another, we were able to define some clusters less dependent on the minutes played by a player. We believe that such a technique can prove to be helpful when analyzing a dataset inconsistent as ours, where both superstars and role players are included.
In Chapter 6 we addressed the second aim of this thesis, which is applying Data Mining techniques to each of the discovered clusters. A variety of techniques were applied, starting from a simple linear regression with automatic selection of variables, since with 27 covariates a manual analysis would have been impossible, going to methods such as Principal Component Regression, Random Forest and Shrinkage methods. Overall, looking at this set of techniques it is easy to understand which was our aim: for each cluster we wanted to reduce as much as possible the number of influential variables, in order to understand if it is true that a cluster formed by shooters and a cluster formed by players specialized in assists influence differently the offensive outcome of a game. This point was indeed addressed by our analysis, since the best techniques overall were Principal Component Regression and Ridge Regression, both methods useful for dimensionality reduction. In a similar way, not all techniques were able to obtain meaningful results, as this was the case for Random Forest Regression and Lasso: in the first case, Random Forest Regression interacts badly with particularly imbalanced datasets, while for the latter we can assume that the Lasso penalization factor was too much of a requirement for our case study, due to a limited number of samples or to a larger issue, which we want to address now as the biggest limitation for this study.
We have previously hinted towards the fact that the dataset used for this analysis did not include advanced offensive metrics to describe more precisely a player's statline. For the whole duration of the study, this has been the most prominent issue, since advanced metrics include information which would have been tremendously helpful in our analysis. Information on, for example, touches and post ups would have incremented the possibility of seeing a separate cluster inside the traditional centers one we were able to discover with k-means. Similarly, a distinction between pull up and catch and shoot three point specialists would have helped seeing a distinction between more "self-centered" players and ones which are more connected to the offensive structure of a team. Having this metrics would help in a more coherent way to construct an entire team: using the trained models discovered in both Chapter 4 and Chapter 6 it could be useful to try different combination of types of players and see which one min-maxes the offensive capability of the team. It is also an interesting mean of studying successful teams, such as 2017 Golden State Warriors, which included perfectly in synergy players in positions not traceable to the traditional ones. For our study, we were limited due to the fact that, as of right now, most of these datasets are behind paywalls, and if that's not the case, the owners (such as the NBA itself) do not authorize the download of any data for personal use. But if it was possible to obtain such metrics a much more deep and precise analysis could be carried on without doubts, and this is the case since a general distinction was clear using the most common metrics. We can conclude on this topic that our study can be seen as a start for a much deeper analysis in NBA players categorization, helped by our conclusions on many topics. The importance of using techniques which mitigate the naturally

imbalanced nature of such realm of study, as well as the lower bound of 9 clusters, from which much more detailed categories can be discovered. Another final point stands in the fact that, for this study, we focused more on stable, reliable techniques and algorithms. An interesting expansion to this study could revolve around utilizing more complex ways to achieve the purposes we intended to achieve.

We believe that the future of sports is deeply linked to the ability of teams and organizations, such the ones in NBA, to gather insightful data that can be used for means such as the ones we discussed here. The specific aim of categorization is still not a topic in the literature for NBA data analysis, while the one of determining the offensive impact of players has seen some interest in recent years. Due to the wide range of applications this instruments may have, we believe that interest on this topics will grow more and more in future years, and will become a standard in the NBA of the future, which analyses thousands of players from all around the world and needs, for this reason, modern and efficient ways to do so.

References

[1] M. Telgarsky and A. Vattani, "Hartigan's method: k-means clustering without voronoi," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 820–827. [Online]. Available: https://proceedings.mlr.press/v9/telgarsky10a.html

[2] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R.* Springer, 2013. [Online]. Available: https://faculty.marshall.usc.edu/gareth-james/ISL/

[3] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2022. [Online]. Available: https://www.R-project.org/

[4] E. T. Jaynes, *Probability Theory: The Logic of Science*, G. L. Bretthorst, Ed. Cambridge University Press, 2003.

[5] Z. Ghahramani, "Unsupervised learning," *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, pp. 72–112, 2004.

[6] T. S. Madhulatha, "An overview on clustering methods," *CoRR*, vol. abs/1205.1117, 2012. [Online]. Available: http://arxiv.org/abs/1205.1117

[7] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, 2nd ed. Cambridge University Press, 2014. [Online]. Available: http://mmds.org

[8] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means clustering algorithm," *Applied Statistics*, vol. 28, no. 1, pp. 100–108, 1979. [Online]. Available: http://dx.doi.org/10.2307/2346830

[9] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[10] J. MacQueen, "Some methods for classification and analysis of multivariate observations," 1967.

[11] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, vol. 1, pp. 231–240, 05 2011.

[12] D. Wishart, “Mode analysis : a generalization of nearest neighbour which reduces chaining effects (with discussion),” *Numerical Taxonomy*, pp. 282–311, 1969. [Online]. Available: https://cir.nii.ac.jp/crid/1571980075067269888

[13] J. Hartigan, *Clustering Algorithms*. John Wiley and Sons, New York, 1975.

[14] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.” in *KDD*, E. Simoudis, J. Han, and U. M. Fayyad, Eds. AAAI Press, 1996, pp. 226–231. [Online]. Available: http://dblp.uni-trier.de/db/conf/kdd/kdd96.html#EsterKSX96

[15] R. Bellman, *Dynamic Programming*. Dover Publications, 1957.

[16] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*. Academic Press, 2009.

[17] P. Domingos, “A few useful things to know about machine learning,” *Commun. ACM*, vol. 55, no. 10, p. 78–87, oct 2012. [Online]. Available: https://doi.org/10.1145/2347736.2347755

[18] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When is ”nearest neighbor” meaningful?” *ICDT 1999. LNCS*, vol. 1540, 12 1997.

[19] K. Pearson, “LIII. On lines and planes of closest fit to systems of points in space,” Nov. 1901. [Online]. Available: https://doi.org/10.1080/14786440109462720

[20] C. Ethington, S. Thomas, and G. Pike, *Back to the Basics: Regression as It Should Be*, 01 2002, vol. 17, pp. 263–293.

[21] D. Maulud and A. M. Abdulazeez, “A review on linear regression comprehensive in machine learning,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 4, pp. 140–147, Dec. 2020. [Online]. Available: https://jastt.org/index.php/jasttpath/article/view/57

[22] S. M. Stigler, “Gauss and the Invention of Least Squares,” *The Annals of Statistics*, vol. 9, no. 3, pp. 465 – 474, 1981. [Online]. Available: https://doi.org/10.1214/aos/1176345451

[23] H. Jiang and K. Eskridge, “Bias in principal components analysis due to correlated observations /,” *Conference on Applied Statistics in Agriculture*, 04 2000.

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[25] J. A. Aslam, R. A. Popa, and R. L. Rivest, “On estimating the size and confidence of a statistical audit,” in *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, ser. EVT’07. USA: USENIX Association, 2007, p. 8.

[26] T. K. Ho, “Random decision forests,” in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

[27] A. N. Tikhonov, “On the stability of inverse problems,” *Proceedings of the USSR Academy of Sciences*, vol. 39, pp. 195–198, 1943. [Online]. Available: https://api.semanticscholar.org/CorpusID:202866372

[28] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996. [Online]. Available: http://www.jstor.org/stable/2346178