RAPPORT FINAL

Projet 3A

March 12, 2017

Members : Gabriel Azevedo - gabriel.azevedo-ferreira@polytechnique.edu
Gustavo Castro - gustavo.pdcastro@gmail.com
Henrique Gasparini - henrique.gasparini-fiuza-do-nascimento@polytechnique.edu

ÉCOLE POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

CONTENTS

1
TITLE EXTRACTING ASDGDFASGDFASGDFASGHDFAE

2
ABSTRACT

In this work, we analyzed the statistics from the 2015/2016 NBA season using well-known algorithms and topological algorithms, inspired by the work of M. Alagappan [1]. We first used traditional algorithms to understand the general properties of the data, as well as the strength of the actual field positions as determinants of playing styles, and then proceeded to a more complex abstraction given by the Mapper algorithm, implemented in the R package TDAmapper [21].

By analyzing the topological structure of the data returned by the Mapper algorithm and comparing it to the structure obtained in Alagappan's work, we could capture a more comprehensive understanding of playing-styles. By identifying the majority of the field positions that were defined in his paper, we empirically proved the robustness of the Mapper algorithm, as we used a more recent dataset from five years later than his.

3
INTRODUCTION

The goal of this project is to analyze National Basketball Association (NBA) data in order to gain insights about the players play-styles. Another goal of the project is, through topological data analysis methods, try to propose a new classification for basketball playing styles, and for that we based our analysis on the following papers (REF HERE).

The project was divided in three major parts. The first one consisted on retrieving and preprocessing the data, as well as using classic tools from data analysis to extract information from the dataset. We used dimension reduction tools in this phase, in order to better analyze the

data from a visual perspective, as reducing the data dimension into two or three dimensions allows us to visualize properly the data.

In the second part of the project, we used classical clustering methods in an attempt to correctly classify the players accordingly to the currently existing classification system (Center, Power Forward, Small Forward, Point Guard and Shooting Guard).

In the third part, we used a method of Topological Data Analysis called Mapper to try to get a deeper understanding about the players. Mapper is a topological data analysis technique that provides further understanding of the layout of the data than usual clustering techniques. In this section, we tried to reproduce the results of M. Alagappan's work, documented in [REFHERE]. He proposed that there actually are thirteen different play-styles in basketball, instead of the classical five, applying the Mapper algorithm to the 2010-2011's NBA players statistics in order to support this theory. Our goal in the third part of this work is to use data from the 2015-2016 NBA season in an attempt to obtain results similar to Alagappan's, in order to give further support to this new different classification system.

4
EXTRACTING THE DATA

Our first task was to find reliable in-game data about the players. We found that a combination of ESPN's website [2] data and the NBA's website [3] data would suit us best. As neither of these websites provided data for the players' salaries, we retrieved it from a third source of information [LINK HERE]. Our data consists of the statistics from the 2015/2016 regular season and contains features from 476 players. Although the data does not correspond to the same one used in Alagappan's work, by obtaining similarly interpretable results, we could empirically prove the strength of his method.

We chose Python as our main language for the project, as it is one of the simplest languages for managing and processing data, and it already contains a package for the Mapper method as well as many others data science packages. In the final and most decisive part of our project we also used R and the Mapper package for R called TDAMapper [8].

5
ANALYSIS OF THE DATA USING TRADITIONAL METHODS

The whole purpose of this part was to process the data in a way that would be useful for us when we start using the Mapper algorithm. More concretely, we wanted to produce an input csv file that could be directly used in the Mapper algorithm pipeline, as well as use unsupervised methods to better understand our data.

First, we applied some dimensionality reduction techniques to visualize and understand the features importances and correlations. Secondly, we used several feature selection methods to determine a few subsets of features that could produce good results. Finally, we compared the results of several clustering techniques when applied to these subsets in order to determine which subset was the most representative and to understand if our data could be easily split in a small number of clusters.

5.1 Preprocessing the data & reducing dimensionality

Our goal in this part was to pre-process the data in a way that would allow us to later apply various clustering methods. Another goal was to visualize data using dimension reduction techniques. This turned out to have an interesting interpretation and even with only two dimensions, we were able to extract important information about the players.

Our initial dataset contained the following variables:

- GP : Games Played
- W: Wins
- L: Losses
- MIN: Minutes Played
- FGM: Field Goals Made
- FGA: Field Goals Attempted
- FG%: Field Goal Percentage
- 3PM: 3 Point Field Goals Made
- 3PA: 3 Point Field Goals Attempted
- 3P%: 3 Point Field Goals Percentage
- FTM: Free Throws Made
- FTA: Free Throws Attempted
- FT%: Free Throw Percentage
- OREB: Offensive Rebounds
- DREB: Defensive Rebounds
- REB: Rebounds

- AST: Assists
- TOV: Turnovers
- STL: Steals
- BLK: Blocks
- PF: Personal Fouls
- DD2: Double doubles
- TD3: Triple doubles
- PTS: Points
- +/-: Plus Minus
- Salary: Players salaries
- PosicAbbrev: Players positions according to official classification

An interesting remark is that we chose to store in the dataset the logarithm with base 10 of the player salaries, instead of using the actual salaries itself. This choice was motivated by the exponential shape of the salary distribution, showed in figure 5.1. This figures show the graph of salary versus players, first in a linear scale, then in a logarithmic scale. We can see that, except for the fifty first players, the logarithmic scaled data transformed is much closer to linear than the original feature was. Since PCA is a linear method, we decided to use logarithm scaled salaries in our analysis.

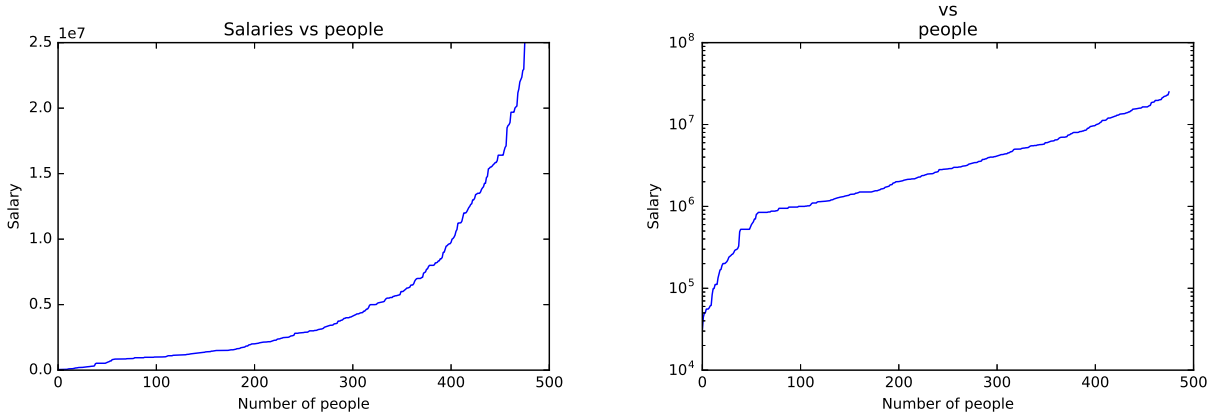

Figure 5.1: Player salaries in increasing order. Logarithm scale in the right

We then performed a principal components analysis (PCA) to two different datasets. They both had the same features from the original dataset, except for the transformation done in the salaries column and the fact that we dropped the column containing the player position labels. The first dataset, herein called *Complete dataset* did not suffer any other transformation. The second one, however, herein called *Time Normalized dataset*, had the time related features divided by the total time of participation of the player in the NBA season, measured in minutes. The names of the features that were normalized are the following:

- FGM
- FGA
- 3 PM
- 3PA
- FTM
- FTA
- OREB
- DREB
- REB
- AST
- TOV
- STL
- BLK
- PF
- DD2
- TD3

5.1.1 PCA - Complete dataset

The PCA performed in figure 5.2 shows that the positions can be partially identified visually even when taking into account only the two main dimensions of the PCA (that is, the dimensions that represent the best the variability in the data). We, however, noticed the second dimension of the PCA is the main responsible for the identification of a player position, while the first component reveals other kind of information. The other components (third and fourth dimensions for example, as it is shown in the figure 5.3) do very little regarding the visualization of the players positions. Figure 5.4 shows the explained variance of the resulting eigenvectors from the PCA.

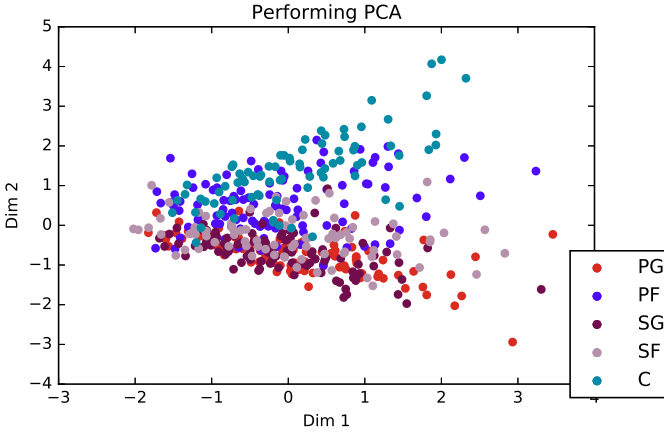

Figure 5.2: PCA over Complete dataset. All positions considered. Dimensions one and two

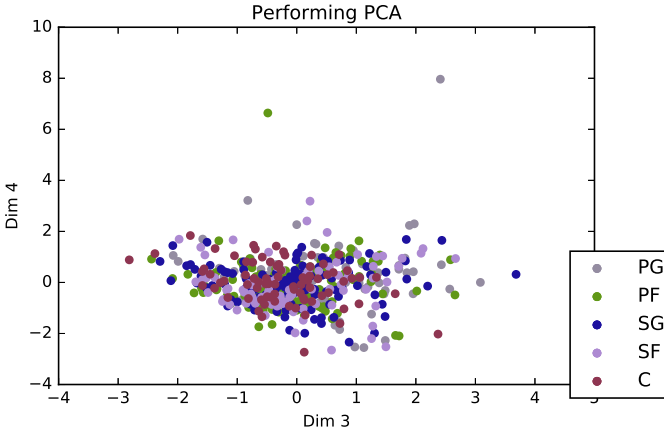

Figure 5.3: PCA over Complete dataset. All positions considered. Dimensions three and four

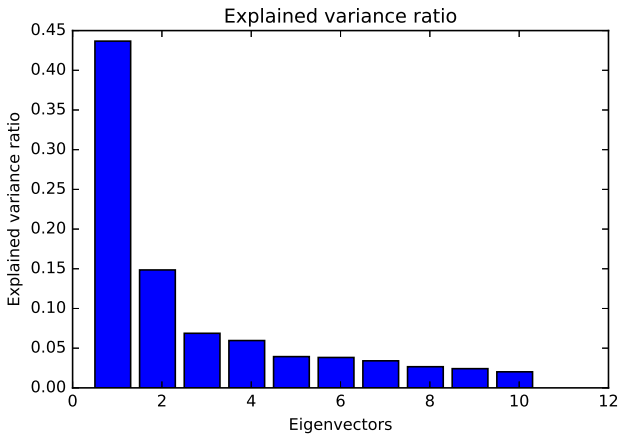

Figure 5.4: Explained Variance. PCA over complete dataset.All positions considered

From the Figure 5.2, we can see that positions where the player tends to attack more (Center and Power Forwards for example) have greater values of Dim 2, while more defensive players (such as Point Guards, for example) have a less important values of the second dimension. This separation becomes more clear when performing the PCA taking into account only players that have extremely different roles in the field, as it was done in the figure 5.5.

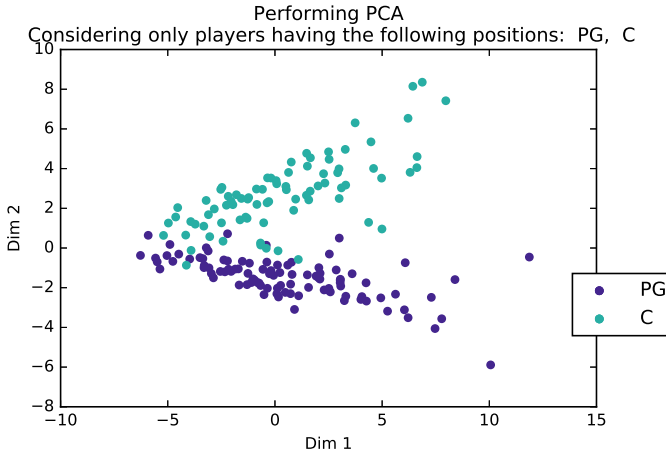

Figure 5.5: PCA on Complete dataset. Positions: Center and Point Gard. Dimensions one and two

In order to see the correlation between the variables, we also plotted the correlation map (also known as variables factor map or circle of correlations), shown in Figure 5.6. It contains, for each feature, its correlation to the dimensions one and two, represented as coordinates x and y, respectively. This graph allowed us to better understand which variables played a more important role for each dimension of the PCA, and thus confirm our analysis regarding the second dimension of the PCA. We can see that the features most correlated to the second dimension tend to concern not only their role as defensive players or offensive players but also

the position of the player in the field. For example, the number of blocks made and rebounds (taken by a player tends to be greater if he occupies positions situated further from the goal, such as Point Guards and shooting Guards. As these variables contribute negatively to the second dimension of the PCA, the least blocks the player does, the greater will tend to be its second composition, and, as those who do more blocks are normally in defensive positions, this further distance from the goal can be related to a defensive way of playing, We can also draw conclusions regarding the first component of the PCA. We can see that the components most related to this component concern mostly the number of points or goals made. We can, then infer that this component is related to what we call the "ability of the player", that is, no matter what is his position on the field, how many points he can make and how he can make his team win.

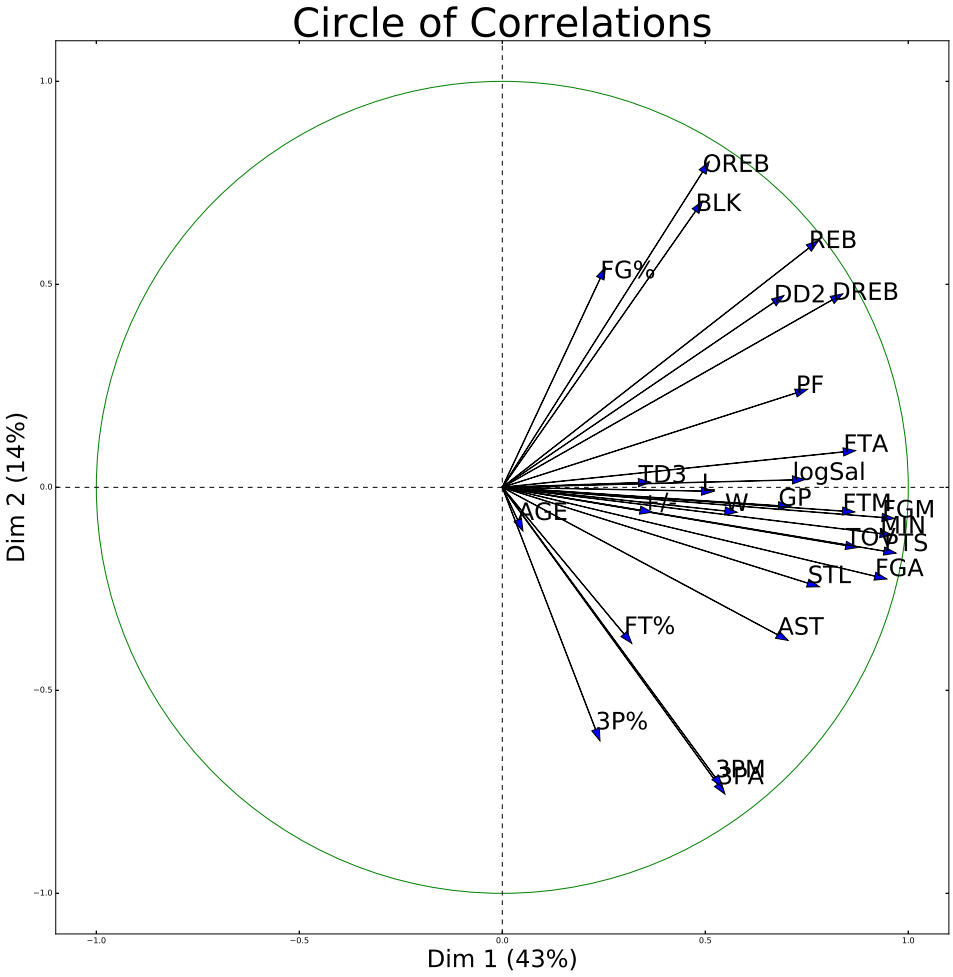

Figure 5.6: Correlation Graph. The coordinates of the vectors correspond to the correlation coefficient between the and the dimension one and two respectively

REVER A PARTE DO RELACAO LOGARITIMA. AS VEZES tirar o grafico e colocar coef de corr de 71% eh melhor já Fiuza - CONCORDO. Tem dois parágrafos gigantes e uma figura pra dizer que a correlação entre os salários e a dimensão 1 é de 71.5% usando a escala logarítimica e que sem usar fica muito pior,

In order to confirm that ”suspeita”, we plotted the PCA coloring the players based on their salaries, making the assumption that the salaries reflect their abilities. We could verify a clear smooth change from low values to high values of dimension one when coloring the graph according to the logarithm of the salaries letting it clear that the dimension one and the feature *log of salaries* are highly correlated. The result of this plot is shown in Figure 5.7. The

correlation coefficient of 71.5% between Dimension one and the salaries (also represented in the correlation map of Figure 5.6) also confirms that this dimensions concerns the abilities of the players. On the other hand the correlation coefficient using salaries in normal scale is , a much less significant value.

We can clearly see that players with smaller salaries have smaller values of dimension one. We can now take another look to the figure Y (PCA colored by positions) and try to explain the reason why, for low values of dimension one, it becomes difficult to distinguish the positions. The explanation may rely on the fact that, the ability of those players could not reach a minimum threshold that allowed to differentiate playing styles. This might also be the reason for, When we analyze the most well paid players (assumed to be the ones with the highest ability levels), the difference of positions becomes very clear.

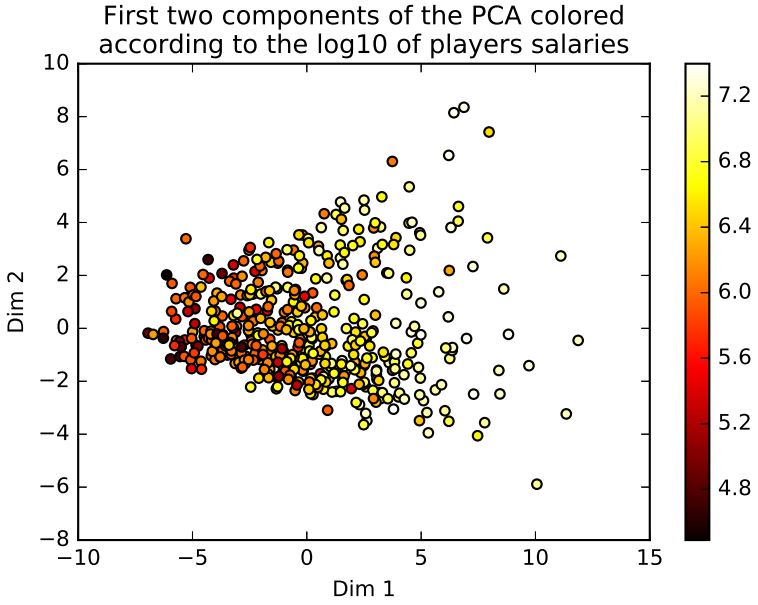

Figure 5.7

5.1.2 PCA - Time Normalized dataset

Similar results were obtained when performing a PCA from the Time Normalized dataset. These results can be resumed on the Figures 5.8, 5.9 and 5.10, which represent, respectively, the PCA itself, the explained variance graph and the correlation graph.

One important remark is that the time normalized data seemed to be better uniformly distributed in space. This opposes to the remarkable degree of separation in the Complete data set, where regions with players with lower values of dimension one (that corresponds to the ability of the player) are much more dense than regions with higher values of dimension one. Thus, we could expect worse results in the clustering analysis, which was the reason that made us choose the unnormalized dataset in our subsequent analysis.

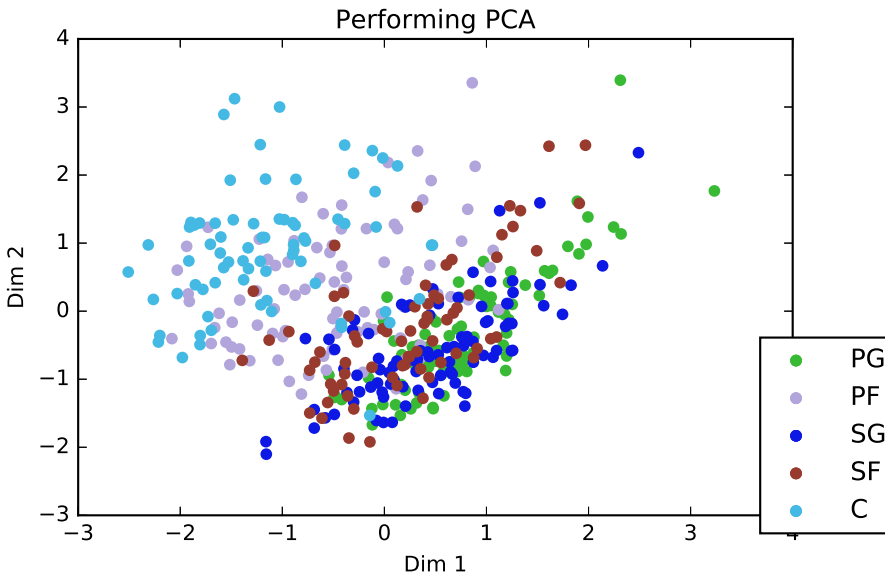

Figure 5.8: Pl

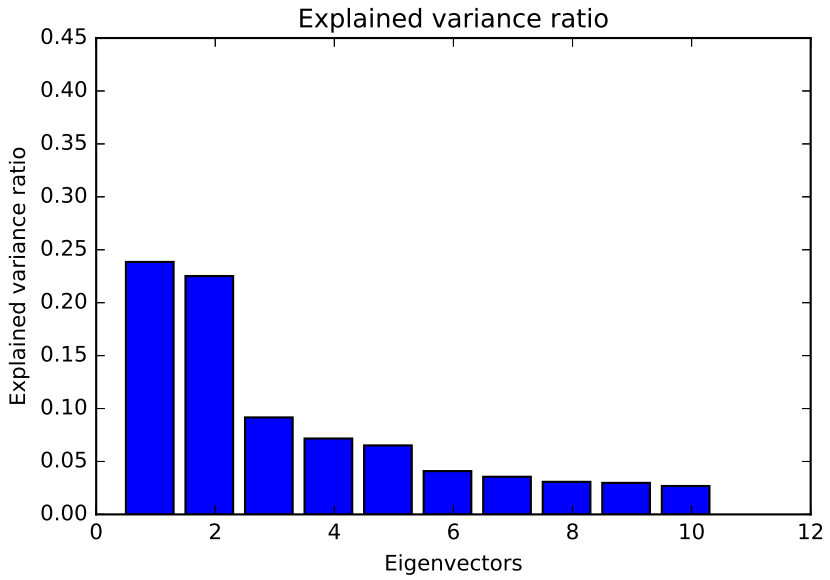

Figure 5.9: Pl

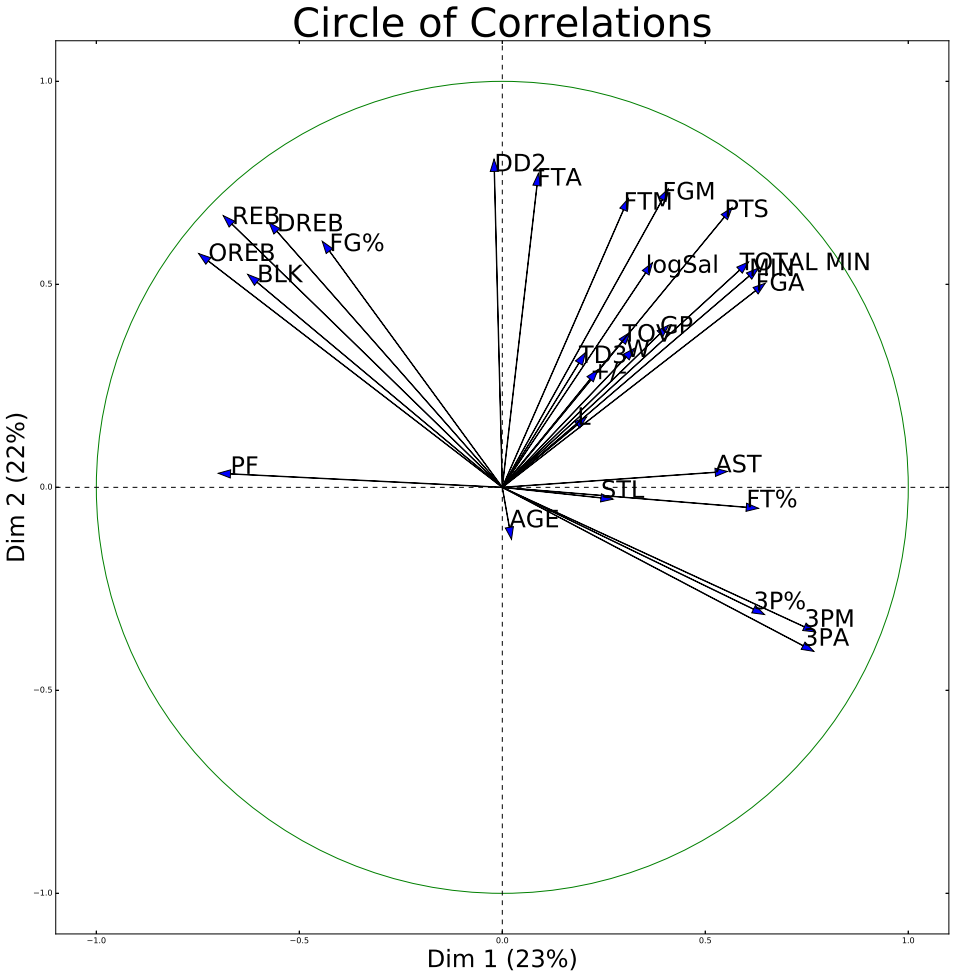

Figure 5.10: Pl

5.2 Feature selection

We aimed to analyze and select the most relevant features for our data analysis, which would first concern the clustering and then the mapper method.

In order to do that, we applied several feature selection methods, implemented in the Scikit-feature library [9] (REF! https://github.com/jundongl/scikit-feature)(o link certo pra citar é o @articleli2016feature, title=Feature Selection: A Data Perspective, author=Li, Jundong

and Cheng, Kewei and Wang, Suhang and Morstatter, Fred and Trevino, Robert P and Tang, Jiliang and Liu, Huan, journal=arXiv preprint arXiv:1601.07996, year=2016) and compared their results in our clustering analyses.

In each method, we specified the number of selected features as seven, for this was the number of features used in Alagappan's work. Moreover, we observed that this was a good balance between not losing too much information from the data and not having to deal with large dimensions counter-intuitive behavior.

5.2.1 Laplacian Score

Selecting features with higher values of the Laplacian score finally produced the best results for unsupervised clustering, particularly when we assumed that there were only three field positions: Forward, Guard, and Center.

The Laplacian score [10] can be seen as an unsupervised metrics of a feature's locality preserving power, i.e., how the values of the feature vary among the k-nearest neighbors of each instance of the data.

We selected the following features:

- AGE
- DD2
- FTM
- 3PM
- TD3
- DREB
- 3P%

5.2.2 Spectral Feature Selection

Spectral feature selection [11] is a similarity-based unsupervised feature selection algorithm and produced particularly good results when we used Gaussian Mixture Models as well as stable results for other clustering methods.

The selected features were: '+/-', u'PF', u'STL', u'AGE', u'BLK', u'OREB', u'TOV'

- +/-
- STL
- BLK
- TOV
- PF
- AGE
- OREB

5.2.3 Multi-Cluster Feature Selection

In multi-cluster feature selection [12], we select features such that the multi-cluster structure of the data is best preserved. The features generated by this method produced reasonably good results using mode-seeking techniques and Gaussian mixture models.

The selected features were:

- OREB
- DREB
- STL
- 3PA
- PF
- +/-
- AGE

5.2.4 M. Alagappan's features

Although Alagappan does not explain how he made his choice of features, we decided to study its robustness and compare their performances in the clustering algorithms that we studied.

We observed that they performed very well when using traditional unsupervised techniques and Gaussian mixture models, also consistently providing good accuracies for other methods.

We recall the selected features:

- REB
- AST
- TOV
- STL
- BLK
- PF
- PTS

5.3 Clustering analysis

We used two main approaches for clustering our data and we could draw different kinds of conclusions for each of them. On the one hand, unsupervised approaches could indicate the degree of separation of our data and how distinct play-styles from different field positions are. On the other hand, supervised approaches analyze rather the quality of the actual labels than the degree of separation of our data. Finally, both approaches allowed us to conclude that the currently adopted field positions do not entirely correspond to the playing-styles and that clustering techniques could not provide a sufficiently detailed understanding of basketball play-styles.

5.3.1 A word on the used evaluation metrics

The most reliable metrics for evaluating the obtained clusters was accuracy. When using supervised methods, this was simply the ratio of correctly predicted labels. For unsupervised methods, we evaluated it as the highest ratio of correctly predicted labels among all possible permutations of the five (or three) labels.

We also used other well-known metrics, such as the Rand index [13] ([14]) and the silhouette score [15] ([16]).

While the silhouette score is an unsupervised measure of similarity of each vertex with vertexes from his own cluster compared to vertexes from other clusters, the Rand index is a measure of

similarity of two data clusterings and was used to compare the predicted labels with the labels from the actual field positions.

5.3.2 Standard unsupervised techniques

In our first analysis, we used traditionally used clustering algorithms, such as KMeans and Agglomerative Clustering.

Our performances were not great, but they already showed that there is some degree of separation between players from different field positions.

The best performances are shown in the table 5.1. Figure 5.11 contains a visualization of the clustering producing the best accuracy.

Table 5.1: Best performances for each evaluation metrics

Metrics	Best score	Score range	Features	Clustering	Linkage
Accuracy	37.8%	[0,100]	Alagappan's features	Agglomerative	Ward
Rand's Index	0.093	[0,1]	Alagappan's features	Agglomerative	Complete
Silhouette score	0.364	[-1,1]	All the features	Agglomerative	Average

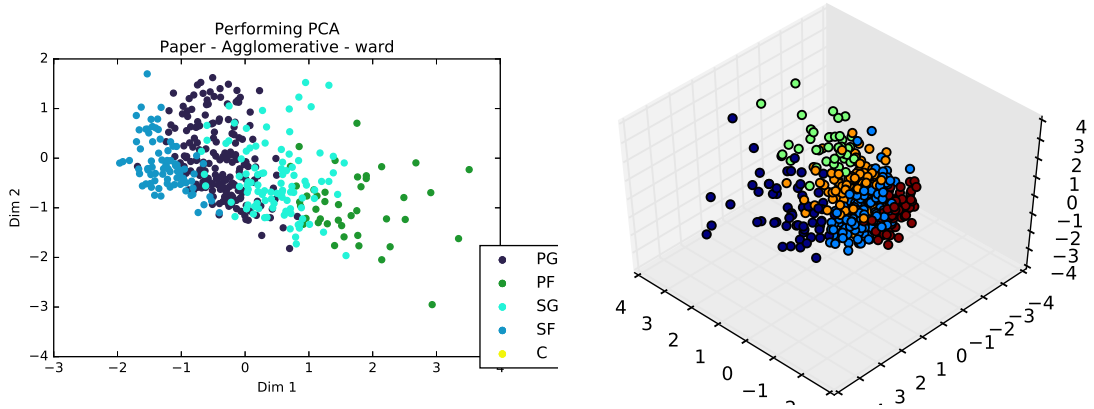

Figure 5.11: Agglomerative clustering using Ward distance and features from the B. Alagappan's paper

5.3.3 Mode seeking techniques

In the mode-seeking paradigm, we assume that data is sampled according to an unknown probability distribution. Therefore, we give greater importance to peaks of density values. In our approach, we used the implementation studied in the session on the mode-seeking paradigm from the course "INF563 – Topological Data Analysis" at Ecole Polytechnique [17] (REF pra um paper de referencia ou não?). We implemented a Hill-Climbing scheme, which is described

in the lab session of the course, and varied our input parameters (*kDensit* and *kGraph*) in order to obtain exactly 5 clusters and better evaluate the results.

Comparing with the results from the previous unsupervised methods, we obtained a better best value for the Rand's index, but worse accuracies and silhouette scores.

The best performances are shown in table 5.2. Figure 5.12 contains a visualization of the clustering producing the best accuracy.

Table 5.2: Best performances for each evaluation metrics

Metrics	Best score	Score range	Features	kDensity	kGraph
Accuracy	34.9%	[0,100]	All the features	12	12
Rand's Index	0.137	[0,1]	All the features	16	8
Silhouette score	0.050	[-1,1]	Laplacian score	24	12

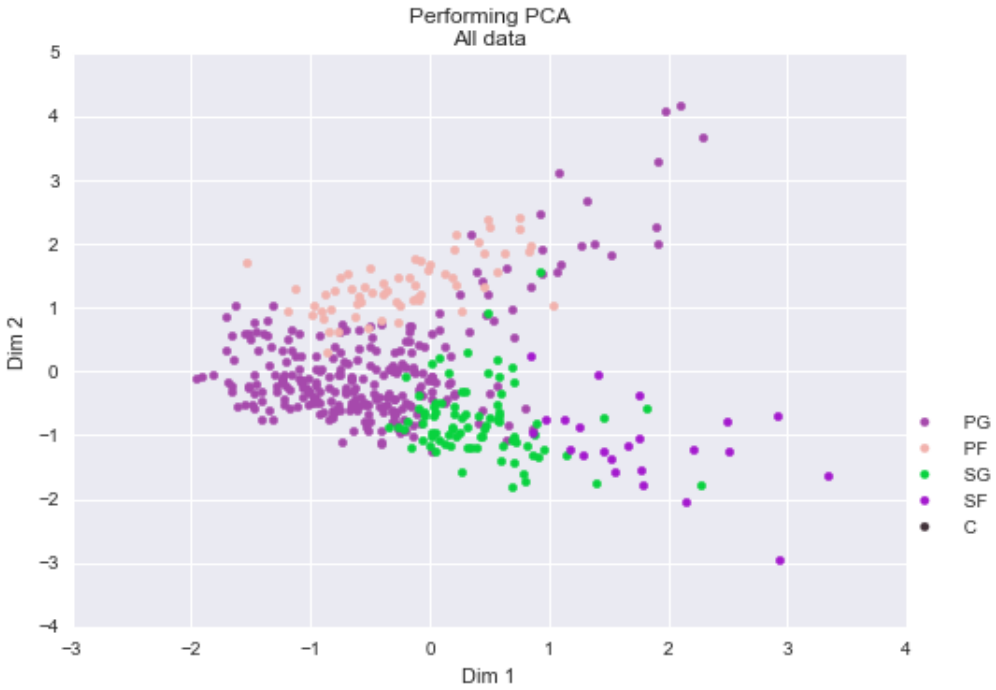

Figure 5.12: Mode-seeking partitioning of the data using all the features

5.3.4 Gaussian Mixture Models

Gaussian mixture model is a probabilistic model that assumes that the data is a sample of a mixture of several Gaussian distributions with unknown parameters (REF possivelmente algum livro/artigo sobre gaussian mixture models). We used the package GaussianMixture from the

scikit-learn library [18] to cluster our data. We used a supervised approach in which we initialize the means of the Gaussians with the means of the classes from the training set [19].

This approach produced the best accuracies so far and other metrics also attained reasonably good values.

This motivated us to analyze other more traditional supervised approaches, which you be shown in the next section.

The best performances are shown in table 5.3. Figure 5.13 contains a visualization of the clusters producing the best accuracy.

Table 5.3: Best performances for each evaluation metrics

Metrics	Best score	Score range	Features	Covariance type
Accuracy	39.67%	[0,100]	Alagappan's features	diagonal
Rand's Index	0.126	[0,1]	Alagappan's features	diagonal
Silhouette score	0.287	[-1,1]	All the features	diagonal

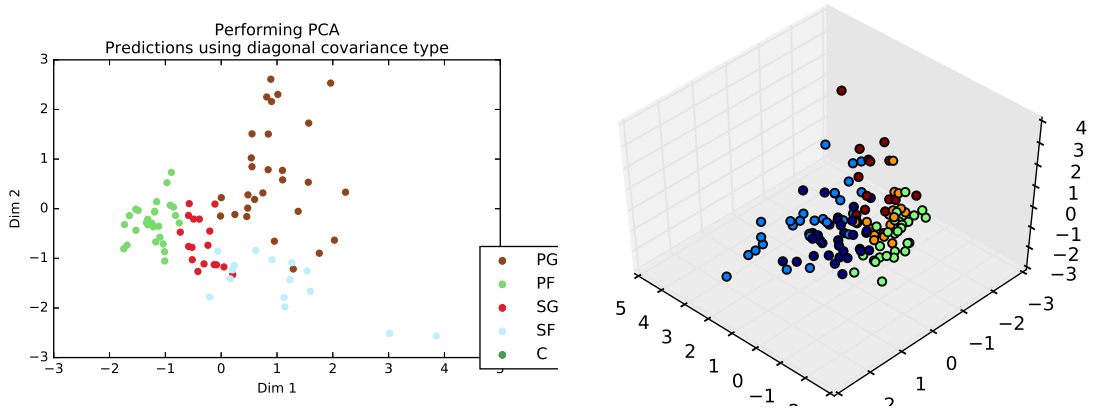

Figure 5.13: Gaussian Mixture Mode using diagonal covariance type and features from the M. Alagappan's paper

5.3.5 Standard Supervised techniques

In order to understand how well field positions were related to playing styles, we decided to use a supervised approach using a traditional machine learning pipeline.

We trained several classifiers and looked for the one that produced the best accuracies. This approach resulted in far better accuracies and gave stronger support for common field positions. We obtained a 63.27% accuracy using a neural networks model.

However, should playing styles from different positions be clearly defined, we would expect even better accuracies. This low accuracy rate could yet be explained by the lack of training data, specially when using deep learning models.

Table 5.4 shows the best accuracy obtained for each classifier we tested and the features that were used to attain this accuracy.

Table 5.4: Best performances for each classifier

Classifier	Best accuracy	Features
Gaussian Process	55.37%	Alagappan's features
Decision Tree	51.24%	Alagappan's features
QDA	55.10%	All the features
Naive Bayes	43.80%	Alagappan's features
Linear SVM	48.98%	All the features
Neural Net	63.27%	All the features
RBF SVM	49.59%	Alagappan's features
Adaboost	54.55%	Alagappan's features
Random Forest	53.06%	All the features
Nearest Neighbors	52.07%	Alagappan's features

5.3.6 Results obtained assuming only three field positions

We could already see in our first analysis that the visualizations generated when partitioning the players in three positions were clearer than when partitioning them in five positions. In fact, Power Forward and Small Forward can be seen as a single position Forward, for they are not easily distinguishable from playing statistics. The same applies for Point Guard and Shooting Guard.

We applied the same methods and obtained far better results, as it is shown in table 5.5

Table 5.5: Best performances for each method

Method	Accuracy	Selected features	More
Common unsupervised	48.95%	Laplacian score	complete linkage
Mode-seeking	48.7%	All the features	*kDensity=20*, *kGraph=7*
Gaussian Mixture Models	67.5%	All the features	tied covariance type
Common Supervised	79.59%	All the features	Neural net

Figure 5.14 contains a visualization of the clusters producing the best accuracy.

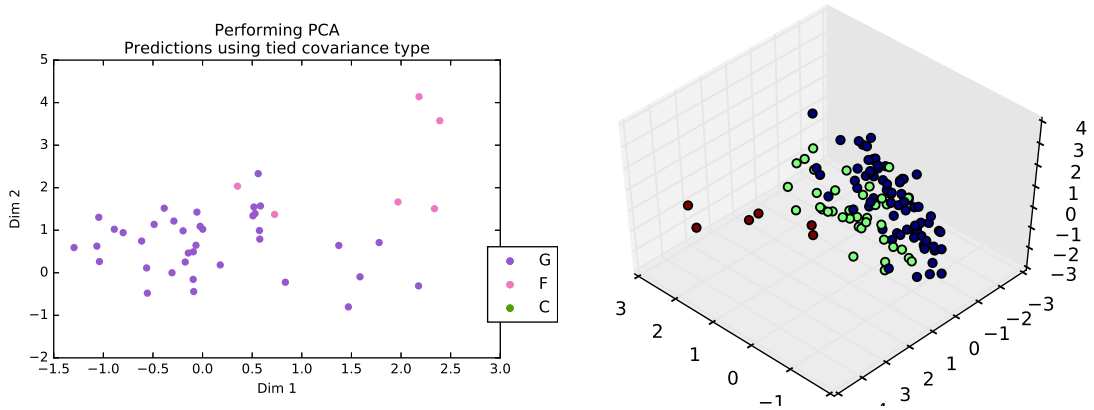

Figure 5.14: Gaussian Mixture Mode using tied covariance type and features from the M. Alagappan's paper to create 3 clusters

These much better performances indicate that traditional division in five positions might be inadequate. The transition from Point Guard and Shooting Guard is not clear and the same applies for the transition from Small Forward and Power Forward. This is another reason for looking for a method that could capture and illustrate these transitions.

We add that a 20% inaccuracy rate is still high and could be explained by a lack of sufficient data, specially when applying supervised clustering. In fact, when using stratified k-fold [20], we observed a sharp rise of prediction accuracies as we increased the number of folds for most used models (in particular, the neural net).

Regarding the choice of features, we observed that using all the features were most often the best choice. However, in order to avoid the well-know "curse of dimensionality", we must choose a small subset of features. By comparing the performances of this different algorithms when applied to different subsets of features, we observed that Alagappan's features produced the most overall consistent results for partitioning players in three field positions.

6
MAPPER

6.1 Theoretical background

In the most commonly used data science algorithms, we approach a problem by testing hypothesis (in supervised methods) or partitioning the data in separate groups (clustering, in

unsupervised methods). Different from these approaches, the Mapper algorithm brings a more insightful understanding of the data, by building a graph that captures the topological structure of the data.

This topological structure is coordinate-free, invariant under small transformations and, in a certain way, can represent complex figures using only a few points and edges, such as in figure 6.15 .

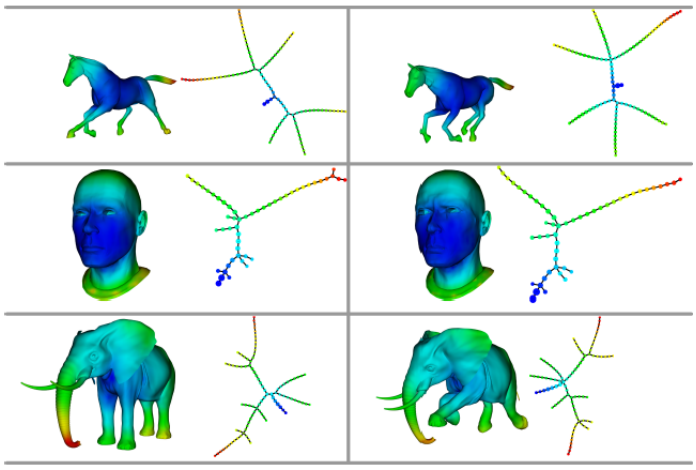

Figure 6.15: Shapes generated by Mapper

6.1.1 The pipeline

The Mapper algorithm must receive three different types of inputs: it takes two resolution parameters (a number N of intervals and p of parameters, also called granularity and gain), a filter function, and a clustering scheme, which is often single linkage clustering.

The pipeline of the studied algorithm can be split in four parts:

First, we compute the filter value of each point in the dataset. We construct N intervals with a uniform p percent of overlap that contain all the values of the filter function.

Then, we clusterize the points in each interval using our input clustering scheme. In the most known implementations, we use single linkage clustering and the input typically is only a parameter that specifies a criterion to the determine the number of clusters in each execution of the clustering algorithm. Each cluster will correspond to a vertex in the final representation

Third, we connect vertexes that share a point from the dataset. These connections exist because the N intervals that we constructed share points in their overlap.

Finally, the graph is usually represented using a force-directed layout algorithm that is coordinate-free and interactive. This part actually does not belong to the Mapper algorithm, but is part of most pipelines implementing it.

Figure 6.16 illustrates a typical pipeline of the algorithm:

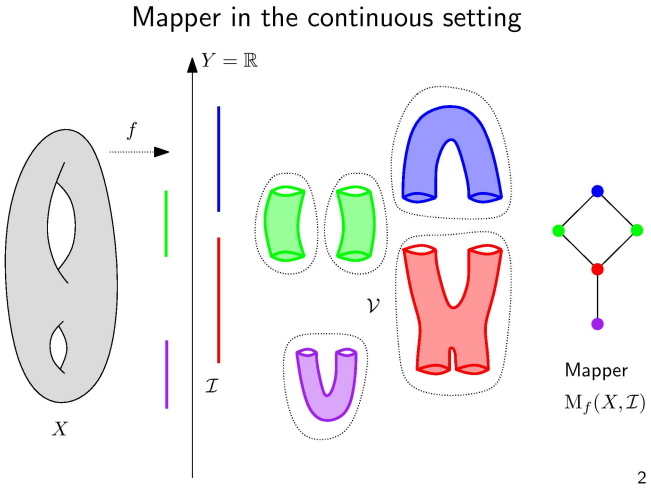

Figure 6.16: Mapper algorithm in the continuous setting

6.2 TDAmapper - R implementation

6.2.1 The input parameters

One of the implementations of this algorithm consisted of an R package called *TDAmapper*, made by Paul Pearson [8].

Although most parameters were already described in the previous section, the filter function and the clustering scheme vary in each implementation. In the case of *TDAmapper*, the default clustering method is single linkage clustering and we only control the variable *num_ bins_ when_ clustering*.

As filter functions, we used the first and second coordinates in principal components analysis.

In order to determine the number of clusters in the third part of the execution of the Mapper algorithm, TDAmapper proceeds as follows: we first construct the single linkage dendogram of the points in the interval using the function hcluster from the package fastclustering(??????). We select the transition values in the dendogram and build a *num_bins_when_clustering* histogram containing these values. We then perform the clustering using the last threshold before the first gap in the histogram.

6.2.2 our approaches to tune the parameters

As explained above (tem que ter exlicado isso eu acho, estah explicado) the mapper algorithm creates a graph that supposedly keeps some of the topological features of the original data, and that could not be identified otherwise due to the high dimension of the dataset. We expected, therefore, that the output graph would show regions, represented by flares or holes, that could be related to playing styles of NBA players.

We applied the algorithm to the *Time normalized dataset* and *Complete dataset* following the notation of section 5. We varied the features, using at first all the features from the dataset and then only M. Alagappan's features.

We extensively varied the choice of parameters and analyzed more deeply those that clearly showed distinct regions. The figure 6.17 represents a part of the various parameter choices we went through. We used Alagappan's features over the *Time normalized dataset* and varied the number of intervals (horizontal axis) and the *num_ bins_ when_ clustering* variable (vertical axes). From right to left, the parameters are 10,20,25,30,35 and 40. From the top to the bottom, the values of the clustering parameter were: 2,4,6,8,10 and 12. The coloring was made according to the average of points scored by the players in each node. We also created other plots were in which nodes were colored according to the most common classical field positions, which helped to create insights about the meaning of the data geometrical structure.

Figure 6.17: TODO

6.2.3 The choice parameters

We chose parameter values that gave a good trade-off between a relatively great number of flares and a small number of connected components. The selected values were:

- intervals : 30 intervals (in each dimension of the filter)
- Percentage of overlap : 50 %
- *Num_ bins_ when_ clustering*: 20

These parameters output the graph of figure 6.18

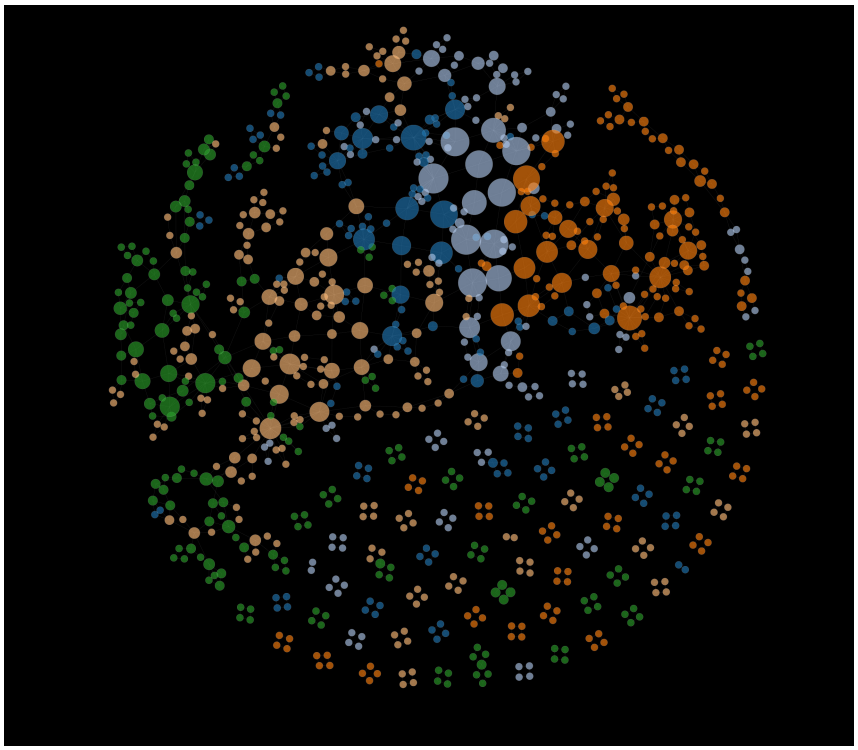

Figure 6.18: TODO

we remarked that, when using number of intervals smaller (?) than 30, the graph became too connected and only a few holes and fares arose. This is illustrated in figure 6.19, where 20 intervals were taken from each dimension of the filter, a overlap of 50% was considered and 20 beens were taken into account. On the other hand, for too many interval divisions, too many connected components are created, which ends up by spliting related players into different components, which "breaks" the meaning of the graph.... We illustrate this fact with, figure 6.20,a graph made from a division of 40 intervals for each dimension (and other parameters unchanged).

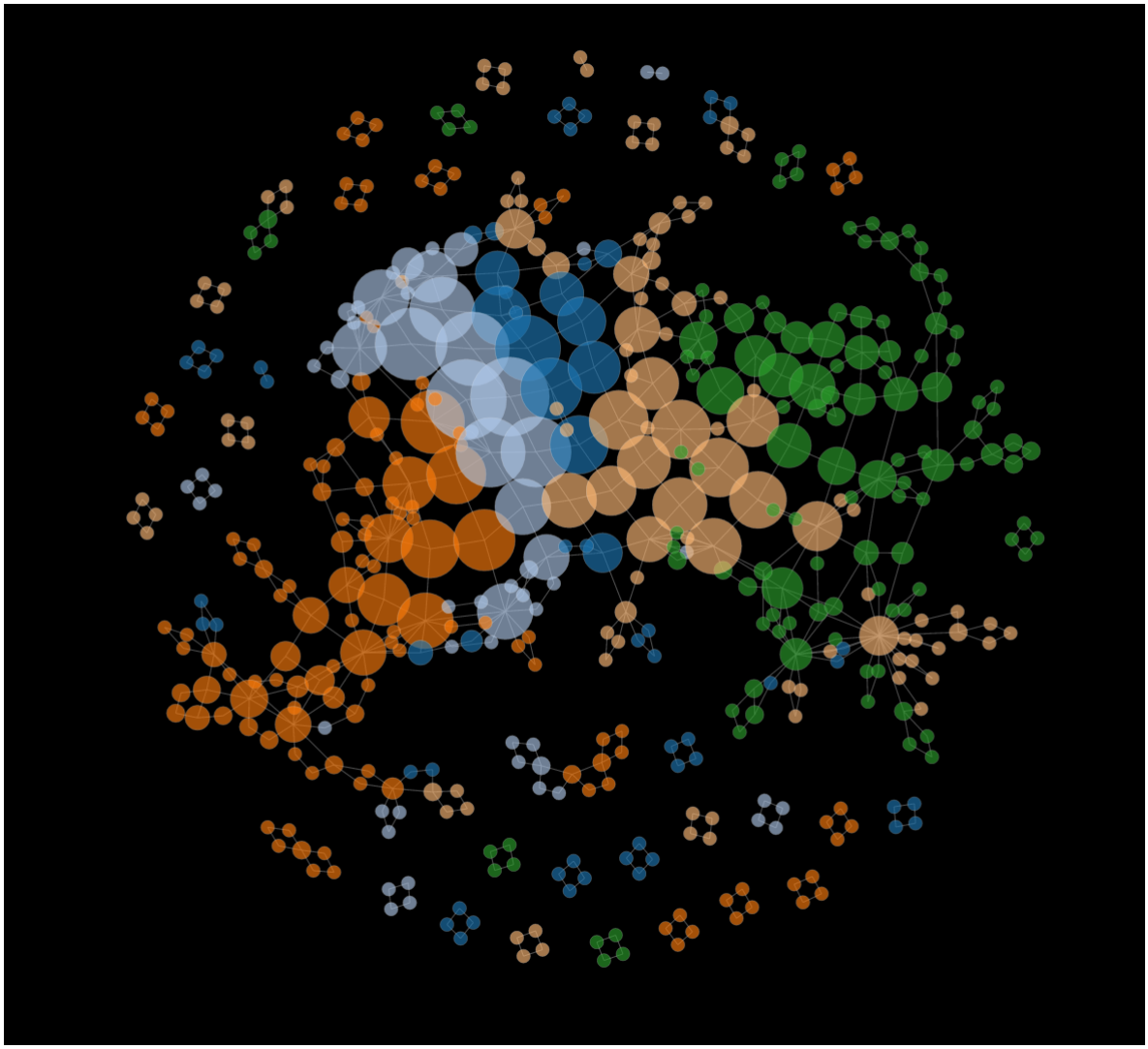

Figure 6.19: TODO

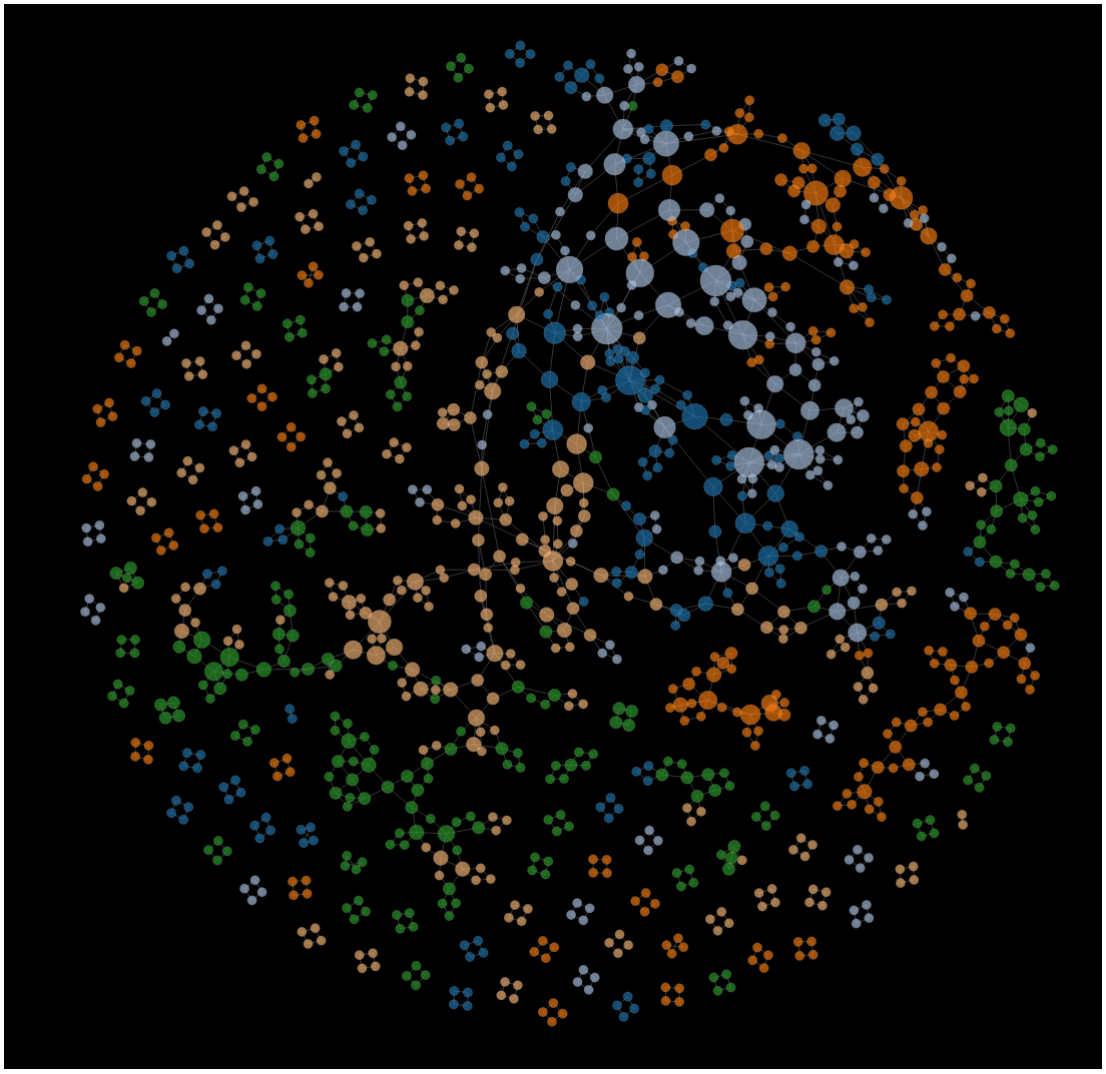

Figure 6.20: TODO

6.3 Analysis

The main goal of the analysis we performed was to be evaluate if it was possible to visualize the same positions that M. Alagappan pointed out in 2012, but now using data from the 2015/2016 season. Our analysis can be illustrated in figure 6.21

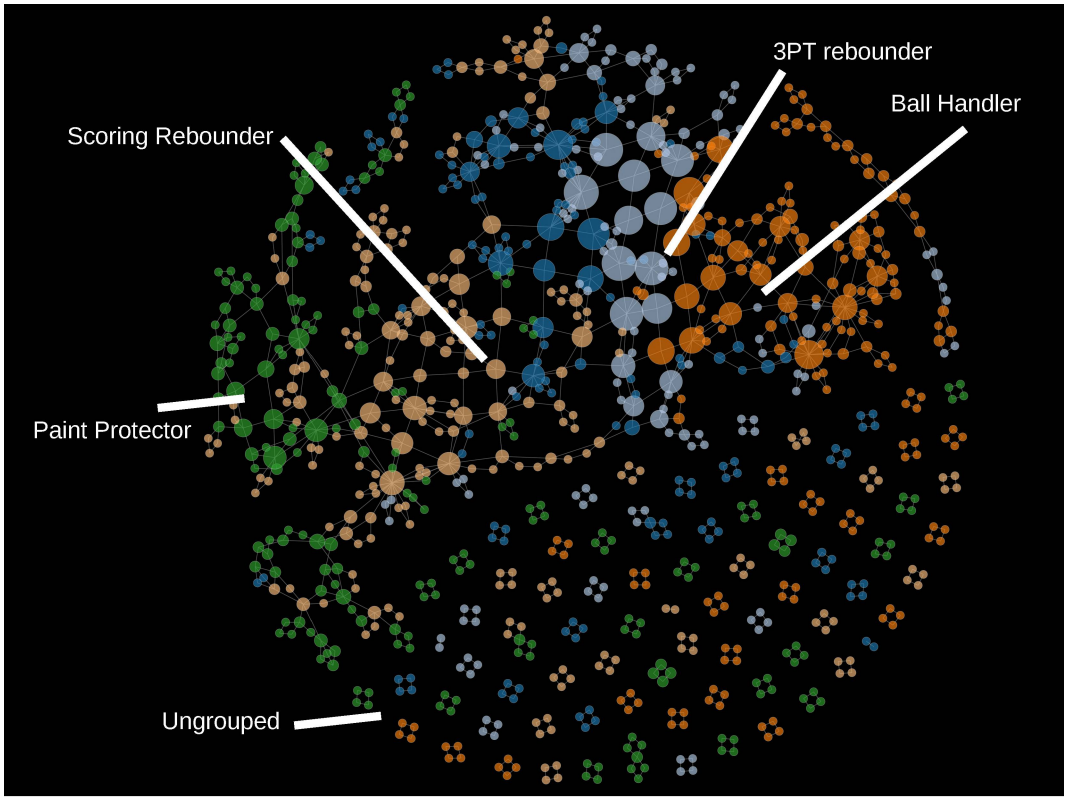

Figure 6.21: Mapper over *Time normalized data*. Using

In our analysis, we compared the players that where representative of each group defined by M. Alagapan, such as Tony Parker, Jason Terry and Tyson Chandler, and checked if the regions they defined (in the analysis done in 2011) were equivalent to those defined from 2016 data. We could remark, as it is shown in figure 6.21, that the major positions of 2011 (shown in figure 6.22) were present in the work: we can see the Ball-Handler, 3PT rebounder, score rebounder and paint Protector positions.

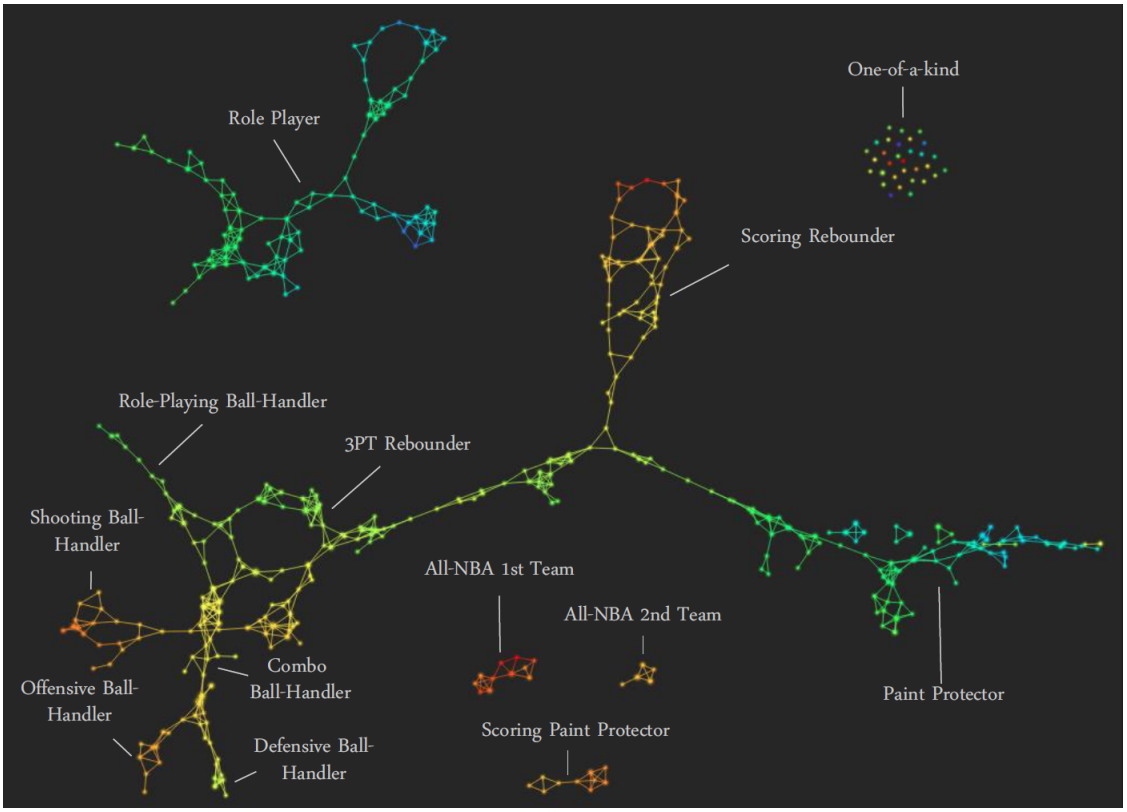

Figure 6.22: Graph made by M. Alagappan in which he proposes a new basketball classification

However, our analysis did not show (even when trying an enormous amount of different parameters) the internal divisions between the rebounder players, shown in figure 6.22. We could clearly see that the players that were in 2011's analysis altogether in the same cluster, in our analysis, were very separate, although coherently placed in the "Ball-Handlers" region. One possible explanation is that such players (for example Jason Terry and Kyle Lorry) may have went through changes in their playing styles, not sufficiently to make them "non Ball-handlers" but to make them more "combo Ball-handlers" than "offensive Ball-handlers" or vice-versa.

Besides, as the two players that were specified to be characteristic of the "Role player position" weren't present at the 2015-2016 season of NBA, it was not possible to identify them in graph.

6.4 Keppler mapper

DEPOIS,

explicar os parametros limitados a 1d. resolvemos usar o angulo, pois, segundo nossa analise da secao 1, ele se relaciona com o estilo de jogo (ofensividade do jeito de jogar do neguinho)

7
CONCLUSION

We consider that our analysis of NBA from the latest completed season (that was the 2015/2016 season) produced very satisfactory results, in that we could identify groups of players that were similar to those found by M. Alagappan in his study.

Our first analysis showed that field positions can be determined from playing statistics with relatively good accuracy but that some positions are less distinguishable, pointing to the necessity of a closer analysis of transitions.

By using the Mapper algorithm through different implementations, we could capture the topological structure of the data and determine new data positions as well as transitioning positions.

A good indicator of the success of our approach is that we could match our groups with the positions found in Alagappan's study in spite of being working with data from different seasons. This fact also confirms one of the key properties of topological methods: they are not invariant under small transformations.

REFERENCES

[1] Alagappan's Work

[2] Espn NBA statistics website

[3] NBA statistics website

[4] Adjusted rand score

[5] Silhouette coefficient

[6] Python Mapper documentation

[7] Mapper Algorithm

[8] TDAMapper (R)

[9] Scikit-Feature

[10] Laplacian Score

[11] SPFS Method

[12] MCFS Method

[13] RI1

[14] RI2

[15] SC1

[16] SC2

[17] TDA-TD

[18] GM

[19] GM1

[20] 3P1

[21] TDAmapper