Clustering Project
Data Analytics Monsoon'21

Shivprasad Sagare
Neel Mishra

September 2021

INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY
HYDERABAD

Contents

1 Introduction

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. The aim is to segregate groups with similar traits and assign them into clusters.

1.1 Clustering Methods

- Density-Based Methods: These methods consider the clusters as the dense region having some similarities and differences from the lower dense region of the space. These methods have good accuracy and the ability to merge two clusters. Example DBSCAN (Density-Based Spatial Clustering of Applications with Noise), OPTICS (Ordering Points to Identify Clustering Structure), etc.
- Hierarchical Based Methods: The clusters formed in this method form a tree-type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category using Hierarchies), etc.
  - Agglomerative (bottom-up approach)
  - Divisive (top-down approach) examples CURE (Clustering Using Representatives), BIRCH (Balanced Iterative Reducing Clustering and
- Partitioning Methods: These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon Randomized Search), etc.
- Grid-based Methods: In this method, the data space is formulated into a finite number of cells that form a grid-like structure. All the clustering operations done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (CLustering In Quest), etc.

We implement the methods like Kmeans, Hierarchical clusteing, and DBSCAN and explain the results in subsequent sections. We also perform data visualization which is explained in section 3.

2 Data

We use the data 'football.csv' that contains the information about 18,207 football players. It has 88 columns or features to describe each player. We use

Numerical	Categorical
ID	Name
Age	Photo
Overall	Nationality
Potential	Flag
Special	Club
International Reputation	Club_logo
Weak Foot	**Value**
Skill Move	**Wage**
Jersey Number	work/rate
Crossing, GKHandling, GKKicking	Body type
Finishing, GKDiving, GKPositioning	Position
Heading, Sliding, GKReflexes	**Joined**
Short pass, Marking, Standing	Loaned_from
Volleys, Composure	**Contract_valid**
Dribbling, Penalties	**Height**
Curve, Vision	**Weight**
FKAcuuracy, Positioning	**LS, ST, RS, LW**
Longpassing, Interception	**CAM, RAM, LM, LCM**
BallControl, Aggression	**LDM, CDM, RDM, RWB**
Acceleration, LongShots	**Release clause**
SprintSpeed, Strength	**LF, CF, RF, RW, LAM**
Agility, Stamina	**CM, RCM, RM, LWB**
Reactions, Jumping	**LB, LCB, CB**
Balance, ShotPower	**RCB, RB**

Table 1: Table denoting the features and their types. The features shown in bold are the ones we identify as originally categorical, but with due pre-processing we convert them into numerical so that we can use them in clustering algorithms to improve the overall quality of clustering.

the pandas library to deal with data as it is well-proven for csv formatted data. It stores the data in special datatype called as DataFrame and makes working on it easier with in-built functions. The first step before working on data is to identify which features are numerical and which categorical. This step is important in deciding which features will be used further for clustering. After manually going through the features, we categorize them as follows. We also identify the possible categorical attributes which can be converted to numerical type using some processing. This will allow us to use more features in clustering algorithms. Below chart summarizes our findings regarding the feature types.

2.1 Data pre-processing

After identifying the features which can be converted to numerical types, we write the preprocessing heuristic for each feature individually and add a new column in the data. For e.g. after processing feature ’wage’, we save it as another column named ’preprocessed_wage’. Following is the sample code to demonstrate how we perform this process on pandas dataframe.

Listing 1: Example of preprocessing value feature and height feature to extract into numerical datatype

```
football['processed_value'] = pd.Series([float(str[1:-1]) //
if str[-1]=='K' else float(str[1:-1])*1000 if str[-1]=='M' //
else 0 for str in football['Value']])

def process_height(value):
    if isinstance(value, float):
        return np.nan
    ft = float(value.split('\'')[0])
    inch = float(value.split('\'')[1])
    inch += ft * 12
    return inch * 2.54

football['processed_height'] = football['Height'].apply(process_height)
```

After creating as many numerical attributes as we can, we move to second step of handling the missing values, if any. We use pandas methods to find the missing value cells and fill it with the mean value of that column. We choose to fill the mean value as it represents the data well for generic exploration.

3 Q1: Data Visualization

Visualization of data is extremely important to identify the patterns and correlations in the attributes. We use seaborn which is a python library based on another library matplotlib to plot the visualizations. We perform all kinds of visualization between numerical and categorical data using univariate distributions, bivariate distributions, categorical bar plots, scatterplots, etc. This helped u in gaining more insights about data. Specific visualizations we tried are explained in subsequent sections.

3.1 Count of players based on Height or other attributes

We plot the count of players on the basis of numerical attributes like Height, Value, and Wage. We preprocess all of these features to normalize them. Histograms are the preferred plots to show the univariate distribution across the dataset. We use seaborn distplot function to plot these visualizations.

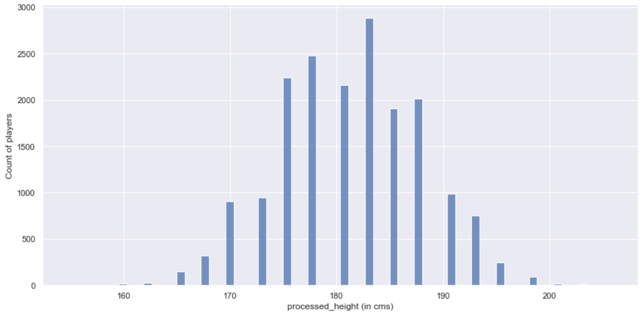


Figure 1: Height feature shows near normal distribution with majority of players centered around the height of 180cms.

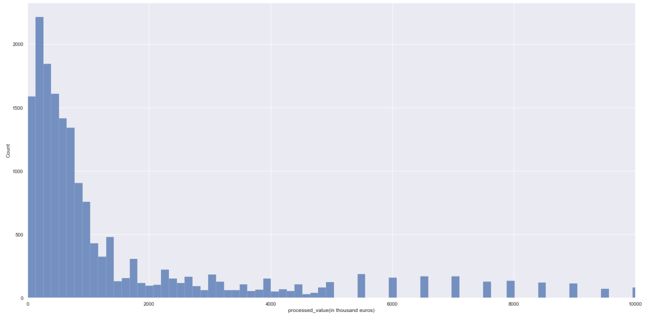


Figure 2: Value feature shows skewed distribution with majority of players having a low value.

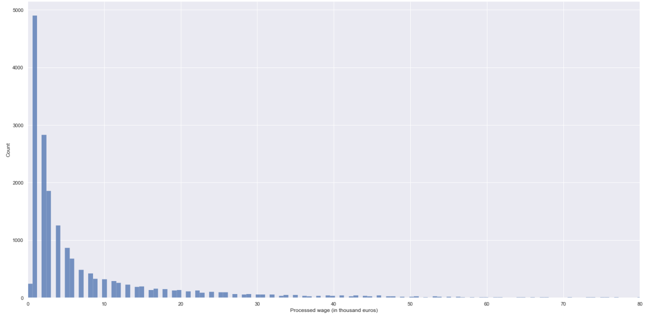


Figure 3: Wage feature shows skewed distribution with majority of players having a low value.

3.2 Distribution in clubs/countries according to some attributes

It is interesting to visualize the behaviour of attribute across the values of other attribute. We can use scatterplot to visualize such relationship if both features are numerical. But if one of them is categrical, boxplot or barplot is the best choice to plot the statistics of numerical feature across the values of the categorical feature. We first of all plot the counts of all the football players across all the nations, and then choose the top five nations for further plotting. This would make our plots easy to visualize.

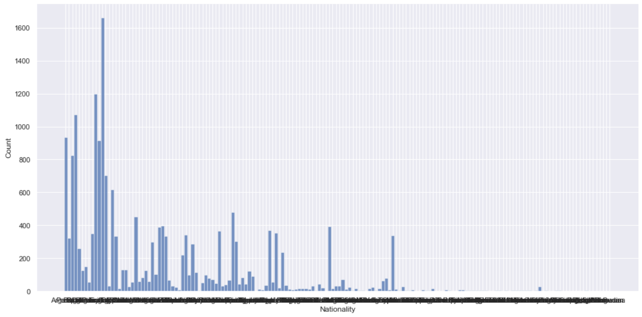


Figure 4: A few nations have a lot more players than the rest all. We choose top 5 out of these for further analysis.

We explore if there exists any disparity in pay-scale based on the nationality of players. For that, we plot the wages of players across the top five nations.

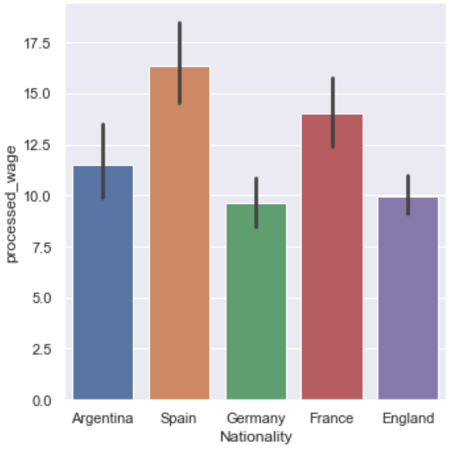


Figure 5: We observe that countries of Spain and France have players with more average wages than Germany, Argentina, and England.

Further, we also plot the overall quality of the player across the nations to see if players from a specific country are better at football skills than the players of other nations.

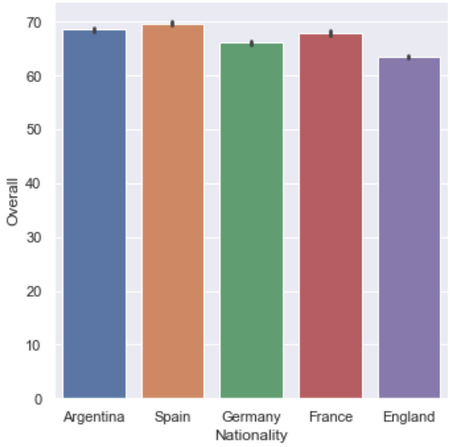


Figure 6: We observe that players from all the countries are nearly at same skills level.

3.3 Features of players according to position

We plot several features of players across all the positions to analyze if some specific position demands a specific set of skills. In the following plot, we show the ages of players specific to the positions.

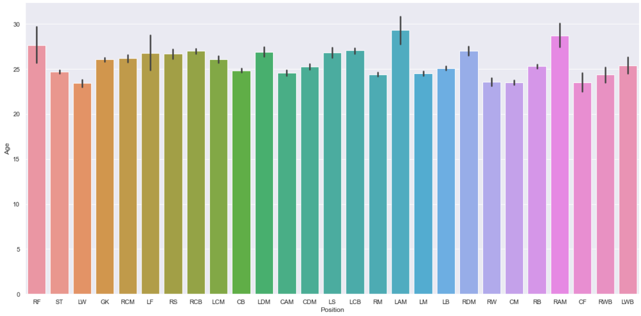


Figure 7: We observe that some positions indeed have players with high mean age than others.

We can infer from above plot that certain position like 'LAM', 'RAM' have higer mean age of players i.e. around 29, and certain position like 'LW', 'CM' have lower mean age of players i.e around 23.

We also plot the feature 'value' w.r.t position, to check if certain position has more or less value attached to it. From the following plot we can infer that certain positions like 'LF', 'RF', 'LAM' have high value associated to them.

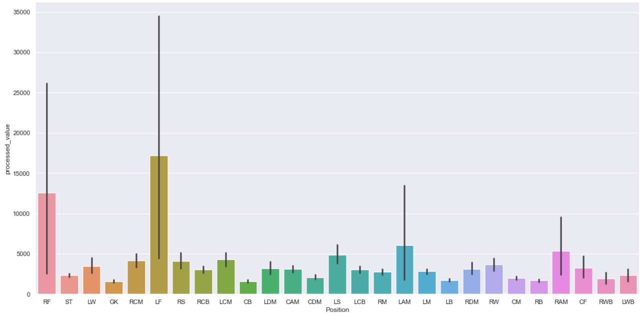


Figure 8: We observe that players assigned to some positions have a very high market value as compared to others.

We observe similar trend as above in case of wages. Players with certain position have higher wages as compared to other team members as can be seen from plot 9.

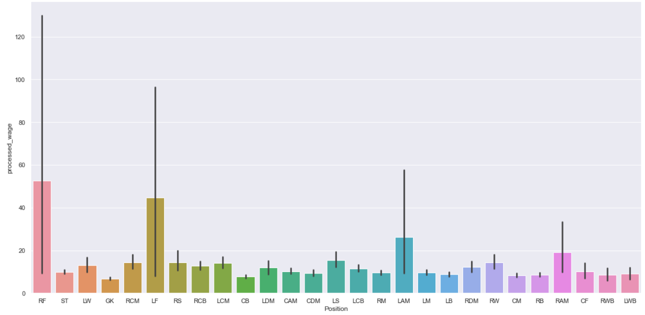


Figure 9: We observe that players assigned to some positions have very high wages amount as compared to others.

3.4 Strategy to pick outliers like Messi and Ronaldo

We adopt a simple technique to detect the outliers in the data in terms of the features like 'wage' and 'value'. We decide to use this as we know that certain players in world are highly rated, with large paycheck and market value. So , we thought that these might be the right features to detect such outliers if any.

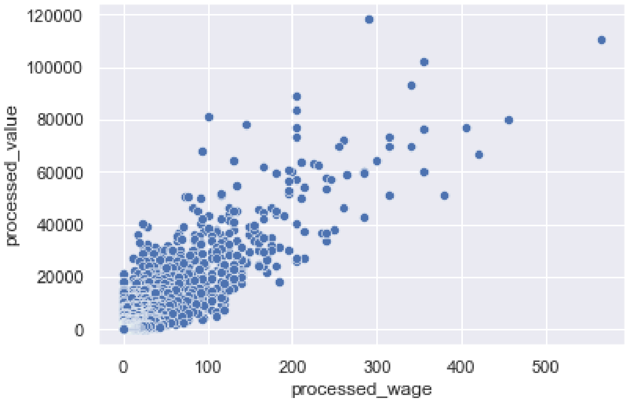


Figure 10: We observe that players in the top right bear high market value and earn high wages as compared to most of the other players. These are the outliers in the data.

From above plot, we can clearly see that among our 18k players, only a handful are located in top right corner of the plot, which means they bear high market value and earn a lot of money as compared to the other players. Outliers like Ronaldo and Messi can be easily found in the top right corner of the plot.

4 Q2: K-means

In k-means clustering we partition the N data points into k clusters, wherein each datapoints belong to the cluster with the closes mean. K-means clustering minimizes within cluster distances.

4.1 Experimental Setup

We have first reduced the dimensionality of the data to 3 dimensions before applying K-means, as it would be easy to visualize the results later. We have then marked our clusters in 3D plots, and have also shown our analysis for each case/experiments.

4.2 For K = 3

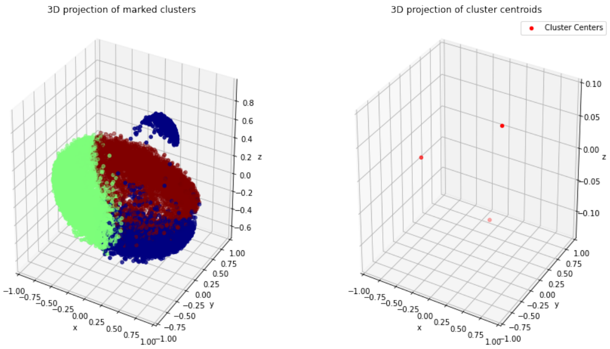


Figure 11: The image on the left shows the marked clusters on 3D projections, while the image on the right shows the respective cluster centers.

4.2.1 Cluster Analysis

- Silhoutte Coefficient

  The Silhouette Coefficient is an example of an evaluation, where a higher Silhouette Coefficient score relates to a model with better defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores

  a: The mean distance between a sample and all other points in the same class.

  b: The mean distance between a sample and all other points in the next nearest cluster.

  Silhoutee result for k = 3 is 0.4118247852805887

- Calinski-Harabasz index

  If the ground truth labels are not known, the Calinski-Harabasz index-also known as the Variance Ratio Criterion - can be used to evaluate the model, where a higher Calinski-Harabasz score relates to a model with better defined clusters.

  The index is the ratio of the sum of between-clusters dispersion and of within-cluster dispersion for all clusters (where dispersion is defined as the sum of distances squared):

  Calinski-Harabasz index value for this k is 13401.493863404461

- Davies-Bouldin index

  The Davies-Bouldin index can be used to evaluate the model, where a lower Davies-Bouldin index relates to a model with better separation between the clusters.

This index signifies the average ‘similarity’ between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves.

Zero is the lowest possible score. Values closer to zero indicate a better partition.

Davies-Bouldin index for this k is 0.9334114998525402

- Determining the most similar attributes of the cluster.

  The detailed analysis of similar attributes is provided in the jupyter notebook, the conclusion that we got was that in our execution second cluster has the players with great values, overall, and better skill sets, balance. They also earn the most.

4.3 For K = 5

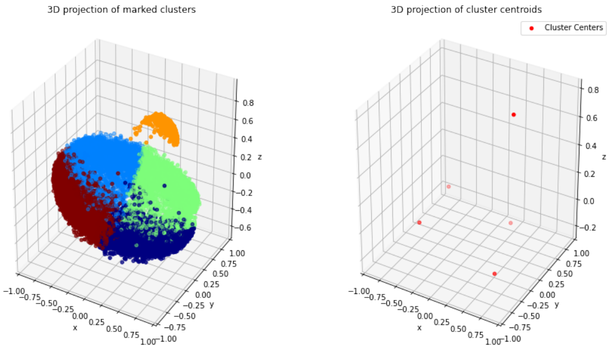


Figure 12: The image on the left shows the marked clusters on 3D projections, while the image on the right shows the respective cluster centers.

4.3.1 Cluster Analysis

- Silhoutte Coefficient

  The silhoutte Coefficient for k = 5 is 0.4951755571768198

- Calinski-Harabasz index

  The Calinski-Harabasz index for k = 5 is 20617.426991338663

- Davies-Bouldin index

  The Davies-Bouldin index for k = 5 is 0.668510025724896

- Determining the most similar attributes of the cluster.

  The analysis is given in detail in the jupyter notebook, the conclusion of the analysis was that the The second cluster has player with higher values, and better skill sets, and the fourth cluster has players with good goal keeping capabilities, but weak overall balance. They also earn average wages.

4.4 For K = 7

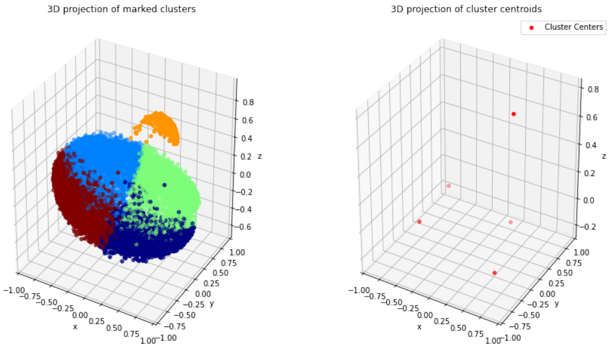


Figure 13: The image on the left shows the marked clusters on 3D projections, while the image on the right shows the respective cluster centers.

4.4.1 Cluster Analysis

- Silhoutte Coefficient

  The silhoutte Coefficient for k = 7 is 0.4377206369522272

- Calinski-Harabasz index

  The Calinski-Harabasz index for k = 7 is 21596.177135115126

- Davies-Bouldin index

  The Davies-Bouldin index for k = 7 is 0.8037557804767238

- Determining the most similar attributes of the cluster.

  The analysis is given in detail in the jupyter notebook, the conclusion of the analysis was that The first, and second cluster has player with higher values, and better skill sets. These players also earn the highest. Example of such players are messi and ronaldo as shown below The seventh cluster has players with good goal keeping capabilities, but weak overall balance. They also earn average wages.

4.5 Finding Optimal value of K using silhoutte, and elbow method

We have plotted silhoutte score for different K, ranging from 2 to 10, and we have found that optimal K occurs at k = 4, although k = 5 is also close, but k = 4 gives the most optimal score as is showin in the figure below.

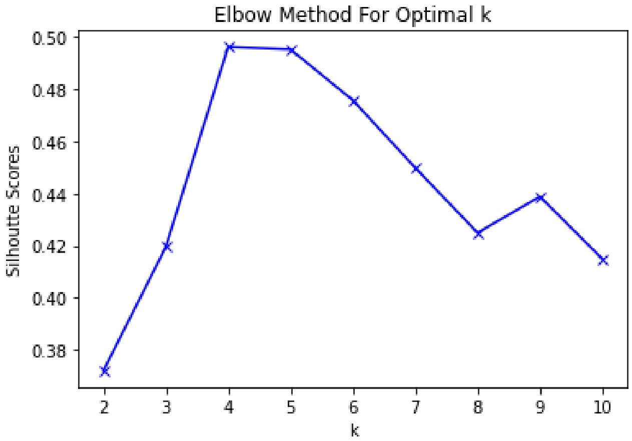


Figure 14: We find optimal K at k = 4, where there is a visible peak.

4.5.1 For optimal k = 4, the analysis

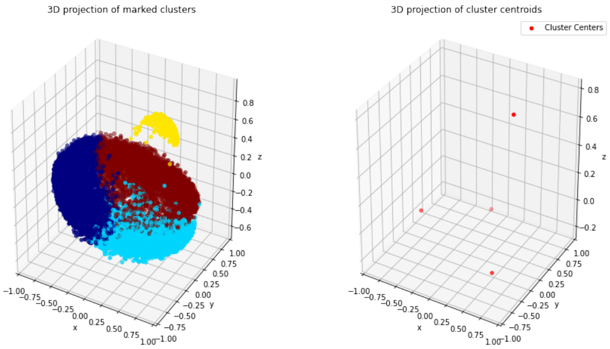


Figure 15: For K = 4

4.5.2 Cluster Analysis

- Silhoutte Coefficient

  The silhoutte Coefficient for k = 4 is 0.4961393255733826

- Calinski-Harabasz index

  The Calinski-Harabasz index for k = 4 is 17085.97315526943

- Davies-Bouldin index

  The Davies-Bouldin index for for k = 4 is 0.685484104505905

- Determining the most similar attributes of the cluster.

The analysis is given in detail in the jupyter notebook, the conclusion of the analysis was that the first cluster has player with higher values, and better skill sets. These players also earn the highest. Example of such players are messi and ronaldo as shown below The third cluster has players with good goal keeping capabilities, but weak overall balance. They also earn average wages

5 Q3: Hierarchical Clustering

Hierarchical clustering involves creating clusters that have a predetermined ordering from top to bottom. For example, all files and folders on the hard disk are organized in a hierarchy. Hierarchical clustering, can be either divisive or agglomerative.

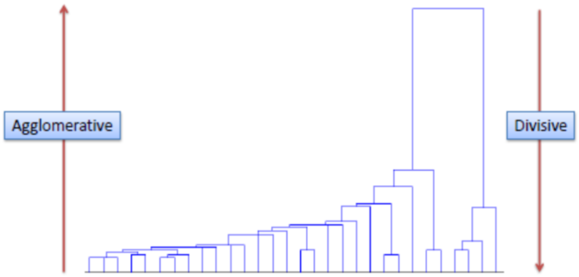


Figure 16: Two strategies of hierarchical clustering. Agglomerative and divisive.

5.1 Agglomerative Clustering

In Agglomerative or bottom up clustering method, we start with individual points as a cluster, then compute the distance between each of the clusters and join the two most similar clusters until we are only left with single cluster consisting of all the points. We decide to use the Python library of sklearn and scipy for the agglomerative clustering use case. Before proceeding with the clustering, we decide which features to use. We extract the numerical features as explained in Section 2.1. An important step here is to scale and normalize the feature values so that they can be compared with each other. This ensures the stability of our clustering algorithm. We use sklearn standardscaler and normalizer for the purpose. Once the data is normalized, we start by creating the dendogram using scipy library. This method internally creates the distance matrix for each data point with other, and creates a linkage matrix based on it. It is fed to dendrogram method of scipy module which took about 1200 seconds for execution on whole dataset. The final dendrogram looked like this.

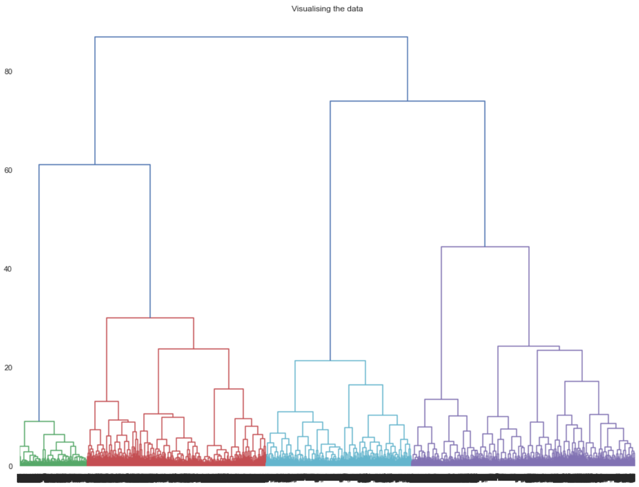


Figure 17: Dendrogram created using agglomerative strategy

Using the dendrogram above, we can infer that optimal number of clusters can be 2 or 4. Similarly, the dendrogrm depicts the optimal clusters using 4 different colors, all of which are of considerable size. We can infer accordingly that this does not predict any group of points as outliers in this plot. We proceed to checking the cluster quality with both of these values. We use sklearn AgglomerativeClustering method to build the clusters with required value of k. The fit_predict method gives the cluster prediction for each of the sample in data. Using it, we plot the scatterplot through which we can see how the clusters are distributed. For the purpose of visualizing a 2-d plot, we have reduced the dimensionality of data to 2 features only. With k=2, we can see the cluster distribution in 18. Similarly, with k=4, the clusters look like in 19. We finally use the silhoutee scores to compare which out of these two clusters is a better choice. sklearn metric silhoutee_score is used for the purpose.

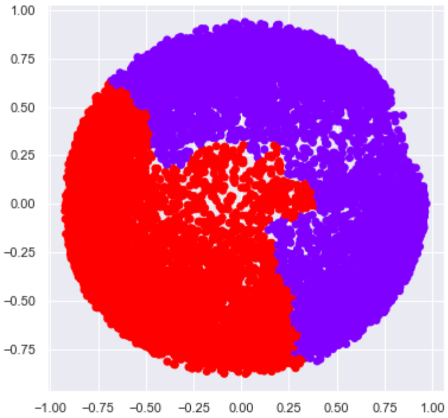


Figure 18: Cluster analysis with number of clusters=2

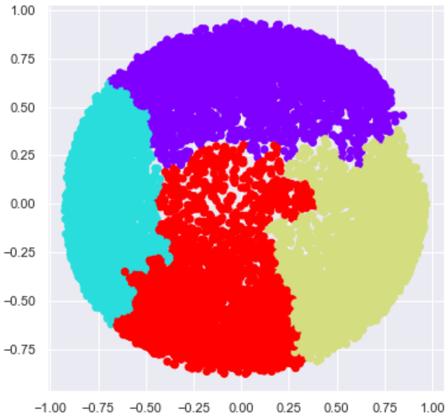


Figure 19: Cluster analysis with number of clusters=4

We perform the comparative analysis of the clustering quality with several hyperparameters in the later section of this report.

5.2 Divisive Clustering

In Divisive or top down clustering method, we assign all of the points to a single cluster and then partition the cluster to two least similar clusters. Finally we proceed recursively on each cluster until there is one cluster for each point. We implement the DIANA algorithm for the divisive clustering, in here. Subsequently, we plot the dendrogram, which is the output of the Diana algorithm. This process also requires calculation of the distance matrix between each row of the data. We use the scipy spatial distance module for the purpose.

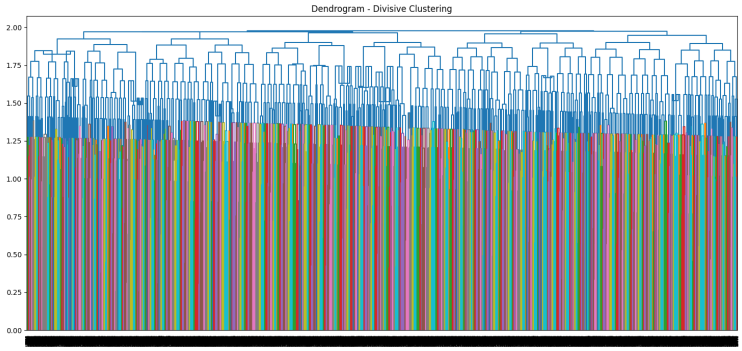


Figure 20: Dendrogram created using divisive strategy

5.3 Analysis

5.3.1 Comparing the clusters

As we observe both the dendrograms, one obtained from agglomerative, and the other obtained from DivisiveClustering, we notice the stark difference in the way that both the algorithms create clusters.

The clusters formed in the first algorithm are seemingly away from each other and well separated. This can be concluded from the long vertical lines in dendrogram, meaning there was a large distance between these clusters before merging them into one. And the clusters formed in the second algorithm are not so well separated as in the first.

This can be determined from the small distance between clusters before merging them into one. The vertical lines in dendrogram are very short in the case of divisive clustering dendrogram.

5.3.2 Cluster Analysis

We do the cluster analysis with n_clusters=2 and n_clusters=4. For both the strategies, we report the evaluation metrics scores and prove that clustering is better with n_clusters=4 in our data.

- The Silhouette Coefficient is an example of an evaluation, where a higher Silhouette Coefficient score relates to a model with better defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores:

  a: The mean distance between a sample and all other points in the same class.

  b: The mean distance between a sample and all other points in the next nearest cluster.

Silhoutee result for k=2 is 0.2083626440116968
Silhoutee result for k=4 is 0.24990771240221585

- If the ground truth labels are not known, the Calinski-Harabasz index-also known as the Variance Ratio Criterion - can be used to evaluate the model, where a higher Calinski-Harabasz score relates to a model with better defined clusters.

  The index is the ratio of the sum of between-clusters dispersion and of within-cluster dispersion for all clusters (where dispersion is defined as the sum of distances squared):

  Calinski-Harabasz index for k=2 is 4814.1320
  Calinski-Harabasz index result for k=4 is 5244.2703

- The Davies-Bouldin index can be used to evaluate the model, where a lower Davies-Bouldin index relates to a model with better separation between the clusters.

  This index signifies the average 'similarity' between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves.

  Zero is the lowest possible score. Values closer to zero indicate a better partition.

  Davies-Bouldin index for k=2 is 1.8664
  Davies-Bouldin index result for k=4 is 1.3048

5.3.3 Labelling the clusters formed

We determine the most similar attributes of the cluster for the purpose of labelling the clusters with meaningful feature names. Hereby, we analyze the clusters obtained with agglomerative clustering, with n_clusters=2.

We determine the mean values of each feature for samples in both the clusters formed and observe them to see the distribution of qualites across the clusters.

From above analysis of mean values of different features across the two clusters, we can easily identify the features which show a lot of variance in the values, and hence are the differentiating factors across the two clusters. Such features with their mean values are as follows:

- Mean values for feature Special across 2 clusters are:
  1735.5880517573644 AND 1392.4243502051984

- Mean values for feature Skill Moves across 2 clusters are:
  2.7640635037166192 AND 1.760922408374905

- Mean values for feature Finishing across 2 clusters are:
  58.014132329999086 AND 26.972016928440837

- Mean values for feature Volleys across 2 clusters are: 53.33587225841975 AND 27.365750101192283

- Mean values for feature LongShots across 2 clusters are: 58.812884280077085 AND 29.664470411049873

- Mean values for feature Positioning across 2 clusters are: 61.722033587225845 AND 32.422572768633046

- Mean values for feature processed_value across 2 clusters are: 3412.0225750206478 AND 918.0205198358414

- Mean values for feature processed_release_clause across 2 clusters are: 6371.322779508805 AND 1922.2846860106215

From the stark variation in the values of these variables, we can label the clusters accordingly. For e.g. the processed value of 1st cluster is 3412 whereas that of second cluster is 918.

Hence we can say that first cluster is of players which high market brand value than second. Similarly, the skills like finishing, volley, longshot, and positioning, show the similar trend.

Cluster 1 has high value for most of these skills indicating that more superior and technically skilled players are included in cluster 2.

To validate our hypothesis, we check the cluster_id labels of Messi and Ronaldo, which both come out to be 0. Both Ronaldo and Messi belong to the same cluster with cluster_id 0. This further validates our hypothesis.

6 Q4: DBSCAN

DBSCAN classifies clusters via the density, i.e it separates clusters of high density from the clusters with low density. The cluster formed by DBSCAN can be of any shape. DBSCAN is heavily domain knowledge based, i.e it's effectiveness depends on the person's domain knowledge, although there are few heuristics to choose the hyper-parameters, if an analyst has the domain knowledge, DBSCAN can be very powerful.

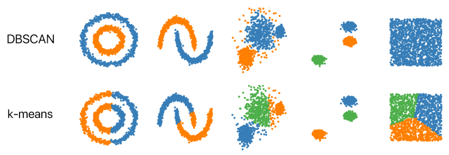


Figure 21: DBSCAN clusters the point based on density.

6.1 Experimental Setup

For the data we use the preprocessed data that we have prepared for the use of all clustering methods. Now after using that data, We have first normalized our data, and then we took PCA to reduce the dimensionality to 3 dimensions. We have done this in order to visualize nicely.

6.2 Choosing the hyper parameter minimum points

To find optimal value of epsilon, and MinPoints we refer to the research paper [1]

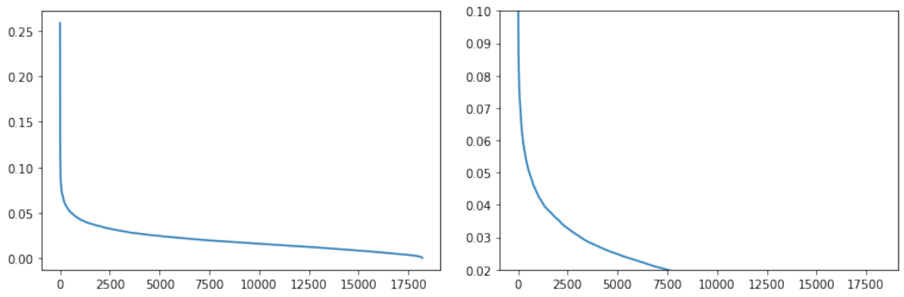
They suggest to keep the min points at 4 for most datasets, and only for those dataset which has a lot of noise it should be kept at 2*dimension. Since we have already reduced the dimension to 3 using PCA, we expect a less noisier data, and hence we can keep the min points at 4.

6.3 Epsilon choice

6.3.1 Approach 1

The paper also mentions clearly that inorder to find the epsilon effectively domain knowledge is required, but without domain knowledge one can try the following.

- We first plot the sorted k-nearest neighbours distance graph.
- We then look for a knee in the graph. This knee can be taken as an epsilon.



(a) Plot for all points

(b) Zoomed plot

Figure 22: sorted k-nearest neighbours distance graph

We can see that there is a slight knee observed in the zoomed graph at 0.04.So we choose epsilon as 0.04

Let's see the clustering below

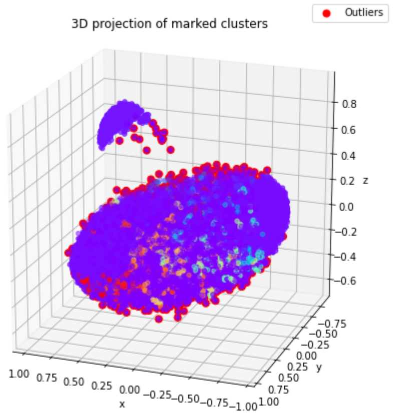


Figure 23: $eps = 0.04$

The resulting clustering is able to classify correct outliers like ronaldo, and messi, but it has low scores, although we have done the choosing of epsilon according to the paper. So we try for our own heuristic approach for the choice of epsilon

6.3.2 Approach 2

Keeping the min points at 0.04 we try to vary epsilon from 0.01 to 0.2. The reasoning behind this is that we have already normalized the data, and the data is dimensionality reduced by PCA so it is less noise prone then at earlier stage.

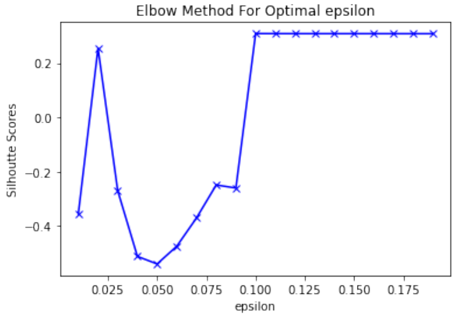


Figure 24: $eps = 0.04$

As shown in the figure we have plotted silhoutte scores for various epsilon. We are looking for the peak elbow like curve, which we observe at 0.1. So taking epsilon as 0.1 we do the clustering.

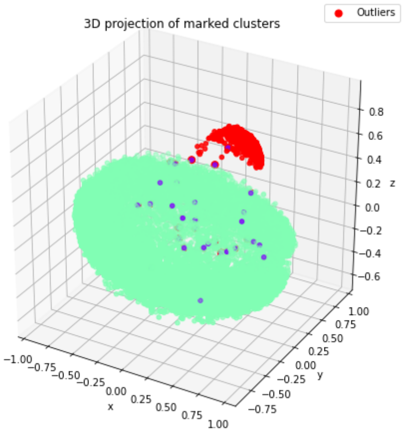


Figure 25: $eps = 0.1$

By doing this heuristic we are getting much better scores, and also is very similar to what DBSCAN would obtain, as it's density based clustering.

6.4 Cluster Analysis

- Silhoutee result for approach 2 is 0.309056445588417897
- Calinski-Harabasz index for approach 2 is 4063.067788329381
- Davies-Bouldin index for approach 2 is 0.7691404120146493

The analysis is given in more detail in the jupyter notebook. The conclusion is that the first cluster has the players who earns the maximum wages, and have way better skills then those in the second cluster.

7 Final Analysis

From our experiments, we observe that k-means and agglomerative strategy leads to very well-separated and comparable clusters. DBSCAN has also given good clusters, but is limited in performace as we don't have a proper domain understanding of the underlying problem, if an analyst has the domain knowledge, they could get very good clusters. Agglomerative clustering creates clusters with approximately similar number of data samples in each cluster, when the number of clusters is set to 4 initially. Similarly k-means algorithm also gives best clustering quality at k=4 as we inferred from the elbow diagram which plots the silhouette score of the clusters according to the value of k. Compared to above methods, the divisive clustering implementation gives not very well-separated clusters with dendrogram plot not very informative. We can conclude from our experiments that kmeans and agglomerative clustering strategies gives us the best insights into the data.

References

[1] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: Why and how you should (still) use dbscan. *ACM Trans. Database Syst.*, 42(3), July 2017.