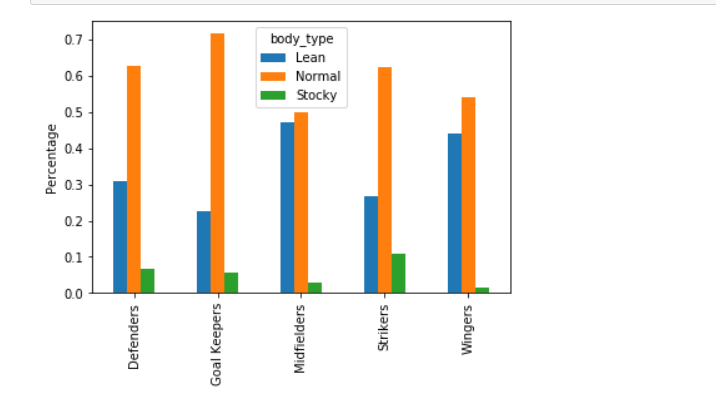
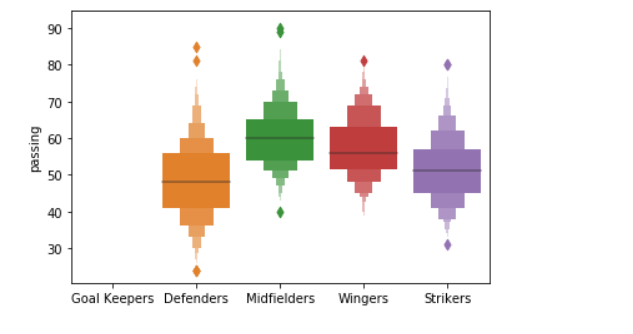
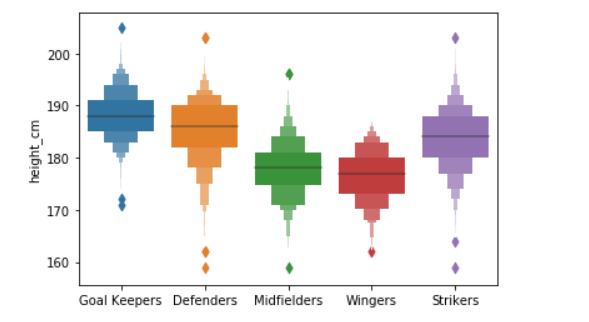
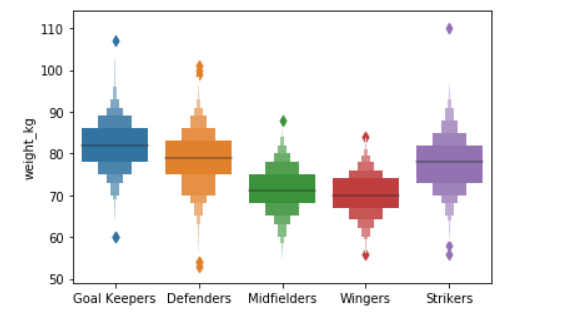
Doc Project

Data exploration:

Here we can see differences between roles and some attributes of players: 

After those plots it’s safe to say that the data is relevant, since the attributes are correlated with the players position.

We can observe that midfielders and wingers are in general shorter and weight less than other players since they need to run the whole pitch and create spaces between the lines. Also, they have a higher passing attribute since its one of their main role in the team.

**Supervised**

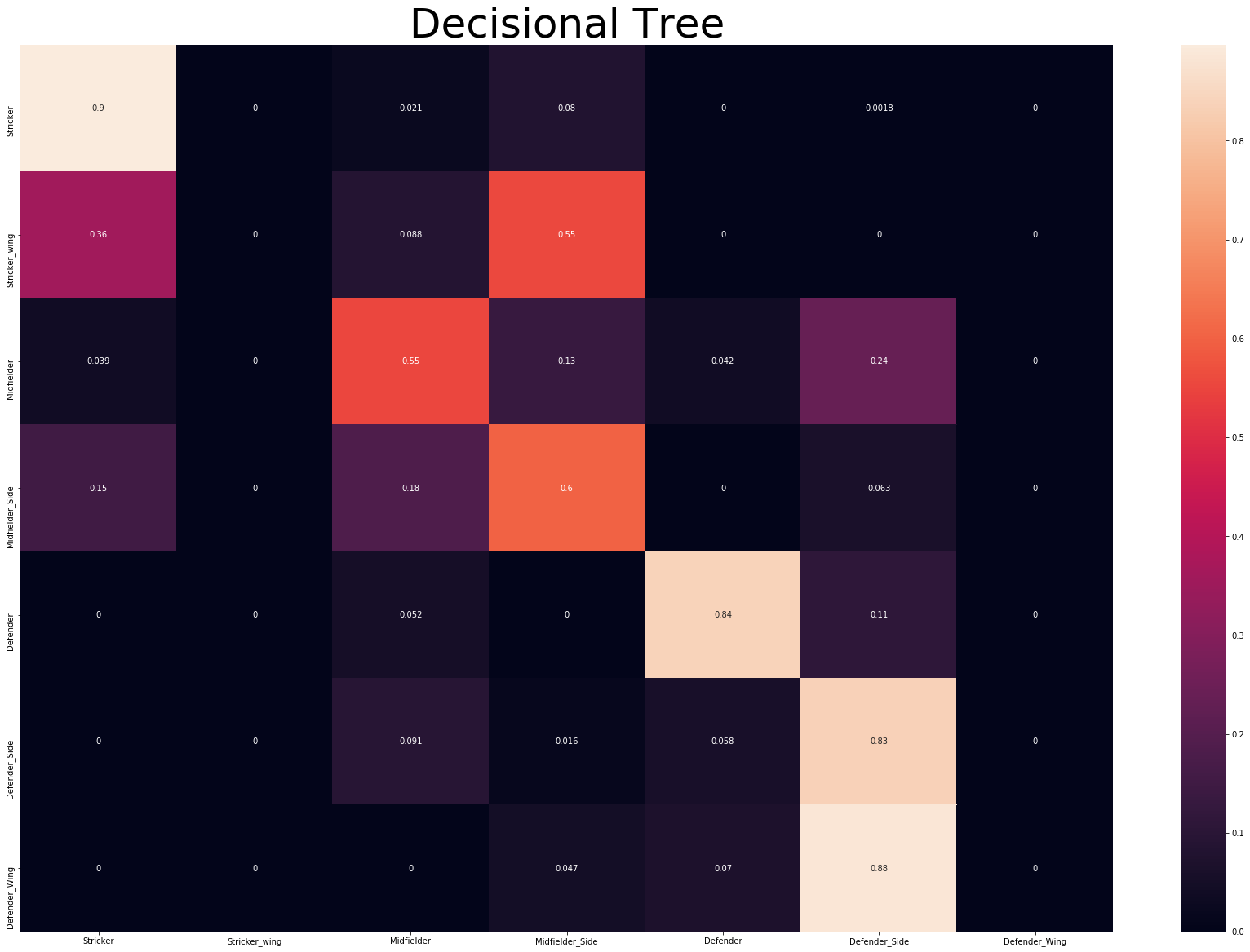
So, after exploring a bit the data set, i will try to implement Decisional Tree algorithm and K-Nearest Neighbor algorithms for the supervised part to predict players position based on their stats.

**KNN**

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

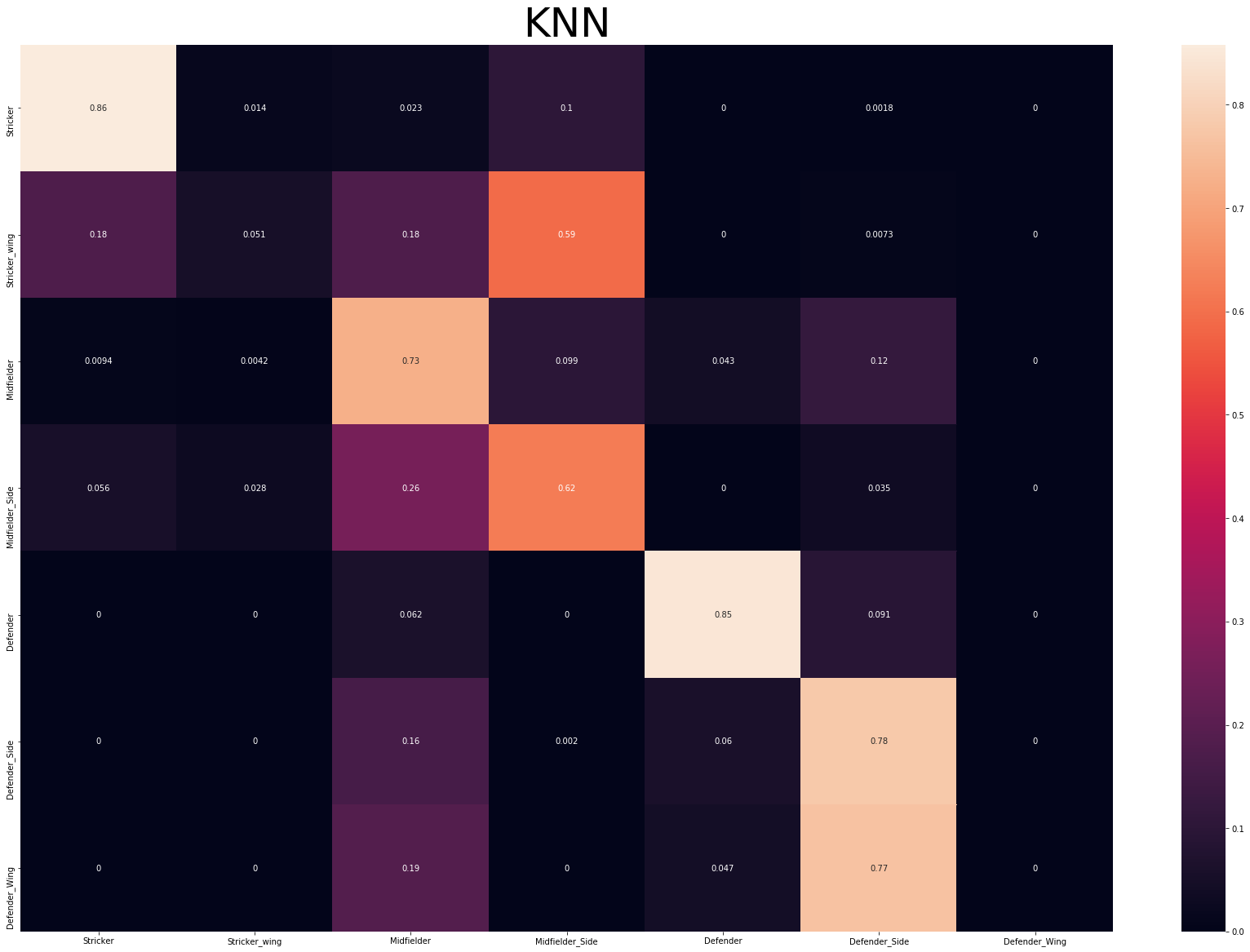
**Decisional Tree**

Decision tree learning is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.



So after running the algorithm on 6 stats we can see its already doing a decent job, scoring around 70%.We can see that the algorithm doesnt have major flows like classifying attackers as defenders or major differences between roles.

I tried playing with the depth of the tree, but increasing the depth, the algorithm would just start to associate players with even more roles(probably since there are so few attributes taken in consideration), also the overall accuracy would decrease.

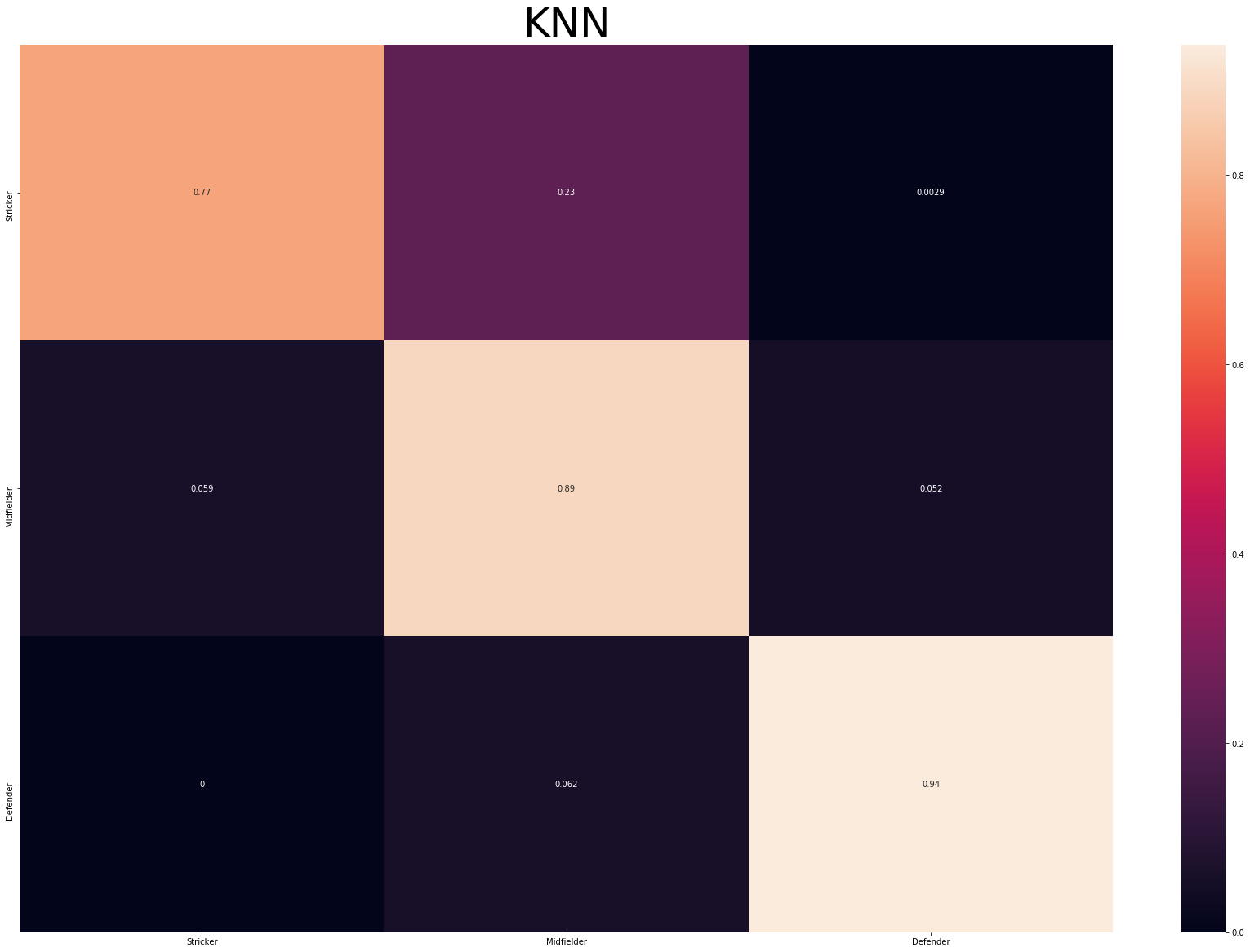
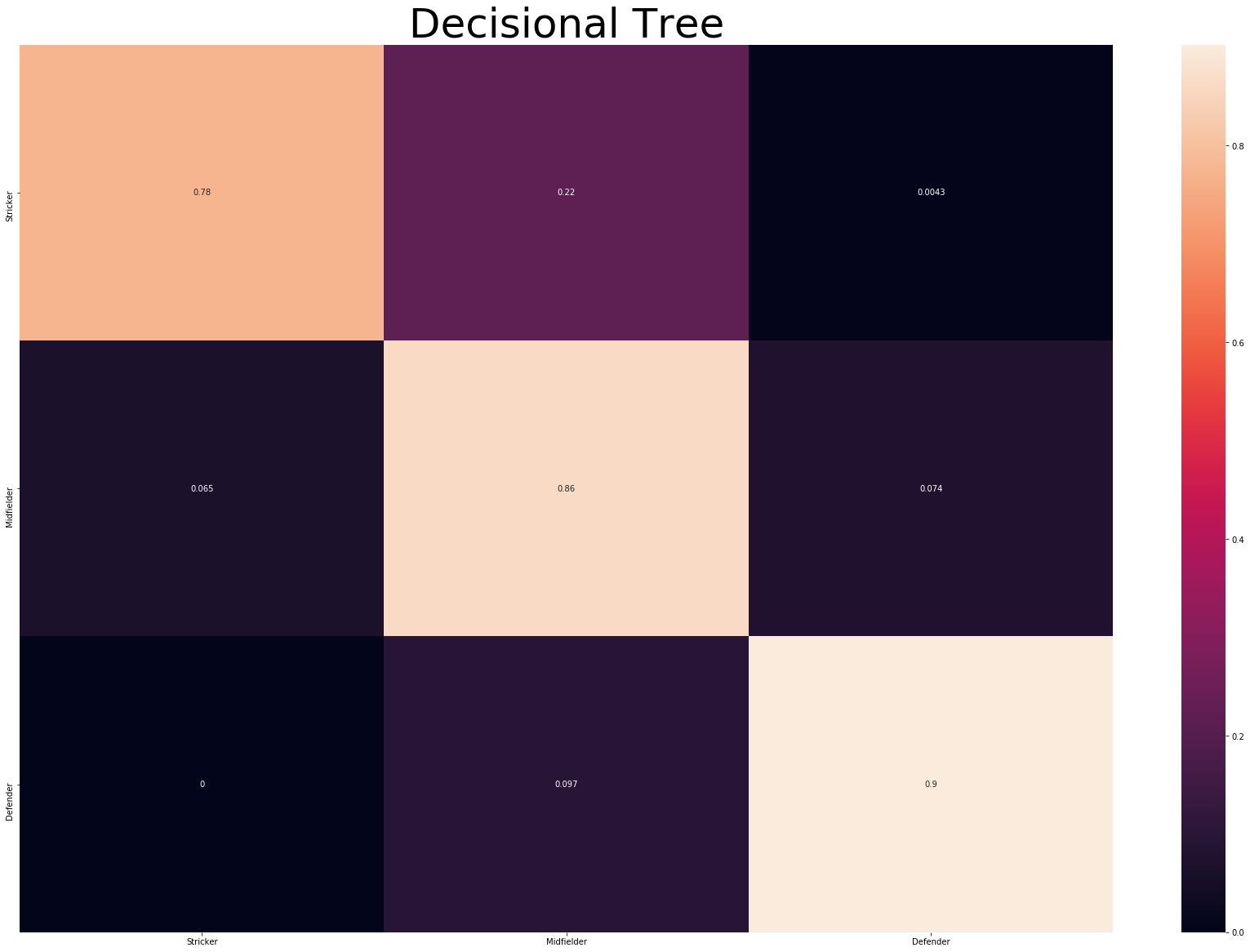


We can also observe that KNN is doing a little bit better than Decisonal Tree, and from the heat map we can also see that it does a better job classifying players around midfield, scoring around 73%.

For KNN, increasing K(nr. of neighbors) would not change much the accuracy and aswell as in decisional tree, it would just try to fit type of players into more roles.

It is normal what the heat map shows since the positions in midfield are versatile and players have similar stats around there.

After adding all players attributes given in the data set, the accuracy of the models raised for both of them around 5-6% with KNN still in the lead. There was a notable difference in their heatmaps. Given more attributes the models managed to correctly classify more roles.

So after this experiment I decided to reduce the numbers of role into 3 main roles: Strikers, Midfielders and Defenders.

We can see that this made quite a difference. Decisional tree scored around 87% and knn scored around 90%. This is mainly because even so there are attacking roles similar with midfielders roles.

I also tried a random decision maker and checked what are the odds of correctly classifying players and given the fact there are 3 classes its around 33% which is not good at all.

**Conclusion**

So, after all the work, we can tell that is possible to classify players into roles based on their attributes. We saw that KNN worked better in all cases than Decisional Tree and we understood the importance of working with the data and getting the correct information and creating the right number of classes.

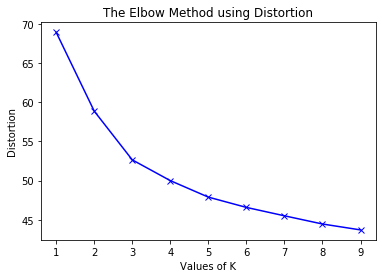
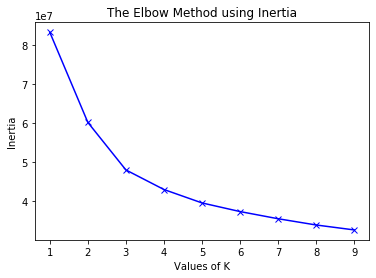
**Unsupervised**

For the unsupervised part i will try to implement K-Means algorithm and DBSCAN.

**K-Means**

K-means algorithm is an iterative algorithm that tries to partition the dataset into Kpre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster’s centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

To establish the optimal number of clusters for K-Means I will use the Elbow method. The method consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use. But first, we're going to drop columns that are not useful for clustering based on relevance like club, nationality and similar features and also based on correlation.

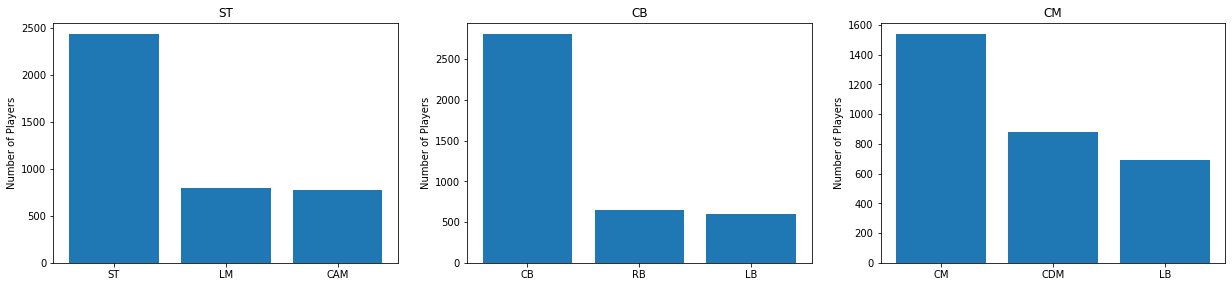


Values of K means number of clusters created.

Distortion: It is calculated as the average of the squared distances from the cluster centers of the respective clusters. Typically, the Euclidean distance metric is used.

Inertia: It is the sum of squared distances of samples to their closest cluster center.

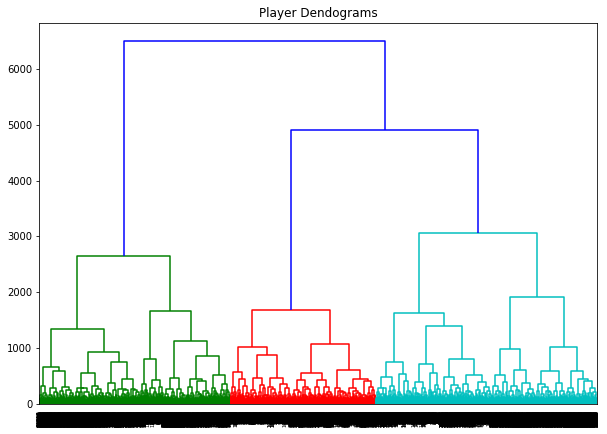
After reviewing the plots generated by the elbow method, we can assume that 3 clusters are more than enough since we can classify players as attackers, midfielders and defenders.



We can see that it did a pretty good job on clustering, every cluster is labeled with the dominant position in it. So we have 3 clusters (CM(Midfield), CB(Defense) and ST(Attack)) and all of them define the 3 positions in football. We can also observe that the CB(Defensive) cluster is the best since the next 2 positions are also defensive ones.

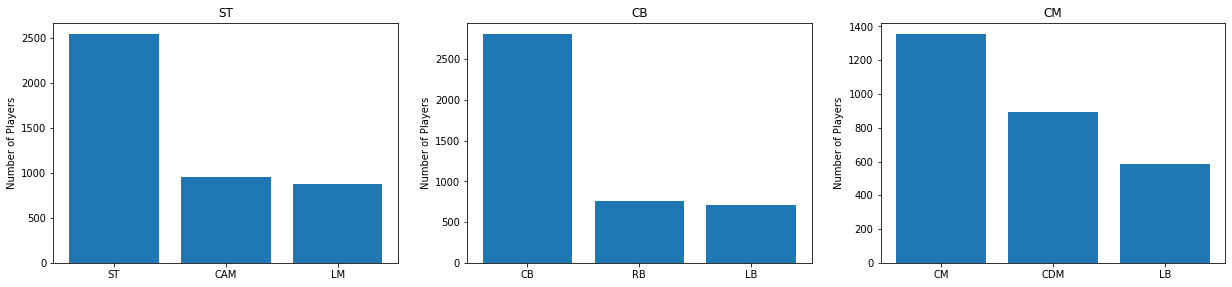
**Agglomerative Clustering**

The agglomerative clustering is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It’s also known as AGNES (Agglomerative Nesting). The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. The result is a tree-based representation of the objects, named dendrogram.



On this dendrogram we can see that it probably starts with defense and attack, then it may be splitting into midfielders and after that it tries to split in all roles. What’s clear is the three main roles in football

A dendrogram is a diagram that shows the hierarchical relationship between objects. It is most commonly created as an output from hierarchical clustering. The main use of a dendrogram is to work out the best way to allocate objects to clusters.



We can see that Agglomerative Clustering produces almost the same results as K-Means (there are some small differences in the numbers of each role). It’s still good since each cluster is dominated by a Defensive, a Midfield and Striker role.

**Conclusion**

So, both the algorithms look for similarities among data and both use the same approaches to decide the number of clusters. Since the data may not be the best cluster-friendly, I think the algorithms achieved what we proposed(to split the players in 3 main roles).