BUS 212a – Analyzing Big Data II, Spring 2019

Project Assignment 4: Clustering

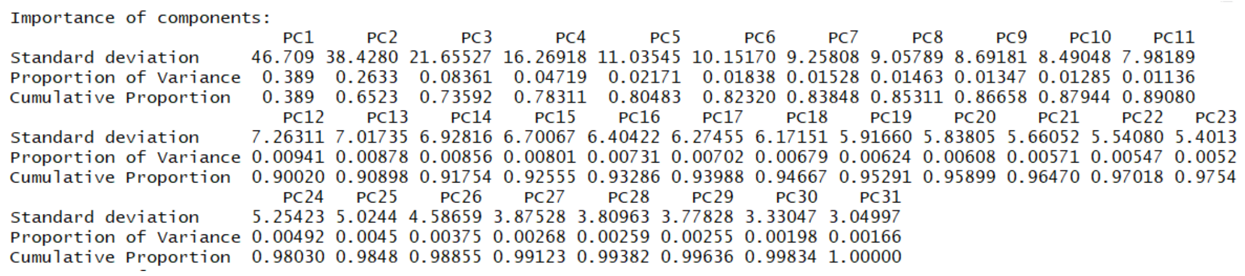
Data Incubator

Part 0. Data Preprocessing and Descriptive Statistics

1. Data Preprocessing

Using the same dataset we chose for the last two assignments (multiple regression and classification), we want to do clustering analysis based on the information of 2019 FIFA soccer players, compare among different clustering methods, choose the most appropriate number of clusters, and convert the result of newly generated clusters to new variables, in order to optimize the performance of our regression model.

Originally, we have 48 variables, each showing the score of a specific capability. After doing PCA for the original variables, we found that the cumulative proportion can only reach 90% when we get to PC12, and we have to get to PC27 if we want a cumulative proportion of 99%. This was too complex and the result was hard to interpret.



```
Importance of components:
                          PC1     PC2      PC3      PC4      PC5      PC6     PC7     PC8     PC9    PC10    PC11
Standard deviation     46.709 38.4280 21.65527 16.26918 11.03545 10.15170 9.25808 9.05789 8.69181 8.49048 7.98189
Proportion of Variance  0.389  0.2633  0.08361  0.04719  0.02171  0.01838 0.01528 0.01463 0.01347 0.01285 0.01136
Cumulative Proportion   0.389  0.6523  0.73592  0.78311  0.80483  0.82320 0.83848 0.85311 0.86658 0.87944 0.89080
                          PC12    PC13    PC14    PC15    PC16    PC17    PC18    PC19    PC20    PC21    PC22   PC23
Standard deviation     7.26311 7.01735 6.92816 6.70067 6.40422 6.27455 6.17151 5.91660 5.83805 5.66052 5.54080 5.4013
Proportion of Variance 0.00941 0.00878 0.00856 0.00801 0.00731 0.00702 0.00679 0.00624 0.00608 0.00571 0.00547 0.0052
Cumulative Proportion  0.90020 0.90898 0.91754 0.92555 0.93286 0.93988 0.94667 0.95291 0.95899 0.96470 0.97018 0.9754
                          PC24   PC25    PC26    PC27    PC28    PC29    PC30    PC31
Standard deviation     5.25423 5.0244 4.58659 3.87528 3.80963 3.77828 3.33047 3.04997
Proportion of Variance 0.00492 0.0045 0.00375 0.00268 0.00259 0.00255 0.00198 0.00166
Cumulative Proportion  0.98030 0.9848 0.98855 0.99123 0.99382 0.99636 0.99834 1.00000
```

Therefore, we managed to find the standard category for the 48 variables given by FIFA website, and categorized the 48 variables into 8 new categories, using rowMeans as the aggregated score for each category. As the number of features are now small enough for interpretation, we will not use PCA before clustering in the analysis below.

2. Descriptive Statistics

(1) Show descriptive statistics for relevant and important variables

	vars	n	mean	sd	min	max	range
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
Release.Clause	1	14743	2.00	4.94	0.01	100.00	99.99
Attacking	2	14743	47.04	8.25	22.17	76.83	54.67
Skill	3	14743	55.43	11.12	22.00	93.40	71.40
Movement	4	14743	66.17	8.76	29.60	92.20	62.60
Power	5	14743	62.03	8.00	33.60	90.00	56.40
Mentality	6	14743	55.46	8.36	31.67	83.17	51.50
Defending	7	14743	51.14	17.83	11.33	92.00	80.67
ST	8	14743	0.13	0.34	0.00	1.00	1.00

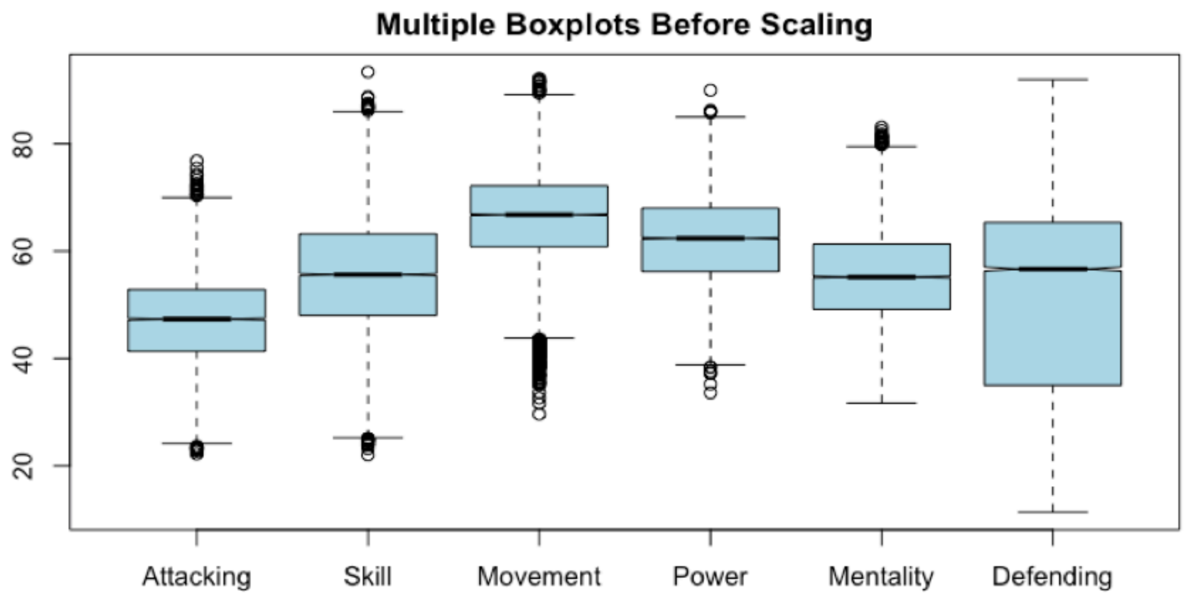
According to the table above, we can discover that the variables have different means and standard deviations. The Release.Clause variable is normalized, while the other variables are based on centesimal measure. In order to avoid the bias caused by different units, we scaled the data.

The table below shows the minimum, maximum, and average (mean, median, mode) and standard deviation / variance of the same variables after normalization. As we can see, the variables now have the same means and standard deviations.

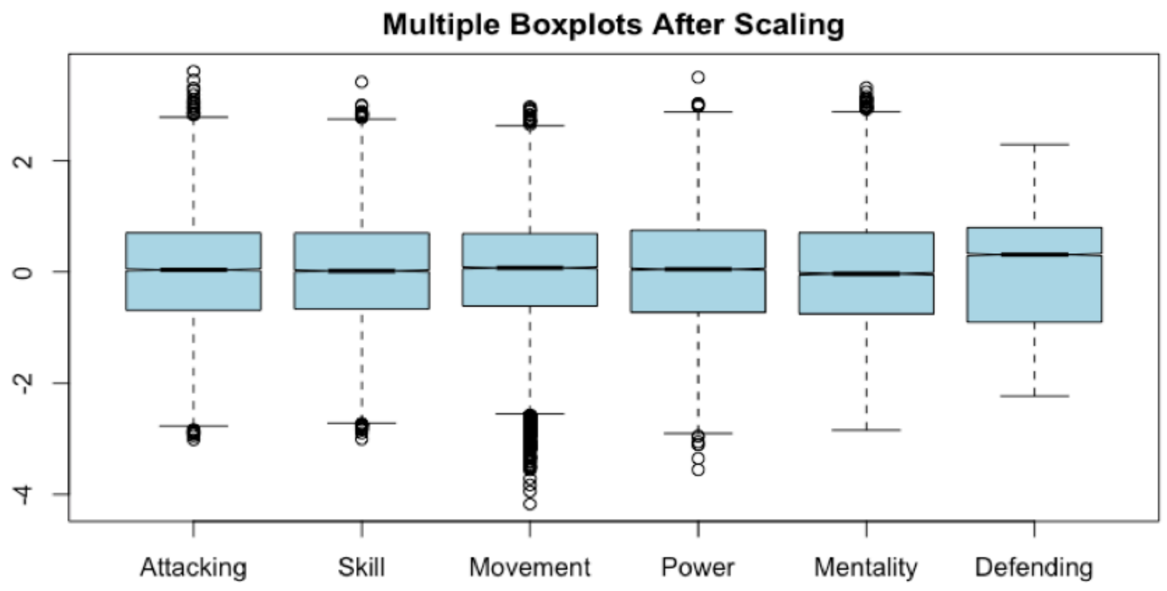
	vars	n	mean	sd	min	max	range
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
Release.Clause	1	14743	0	1	-0.40	19.85	20.25
Attacking	2	14743	0	1	-3.01	3.61	6.62
Skill	3	14743	0	1	-3.01	3.41	6.42
Movement	4	14743	0	1	-4.17	2.97	7.14
Power	5	14743	0	1	-3.55	3.50	7.05
Mentality	6	14743	0	1	-2.85	3.32	6.16
Defending	7	14743	0	1	-2.23	2.29	4.52
ST	8	14743	0	1	-0.39	2.58	2.97

(2) Box-and-Whisker Plots

The boxplots below depict the distribution of the categorized but unscaled data. The variables have different distribution even though they are all based on centesimal measure. The Defending variable has larger range of value than the other variables. Movement, Mentality, and Power have more observations gathered around the means.



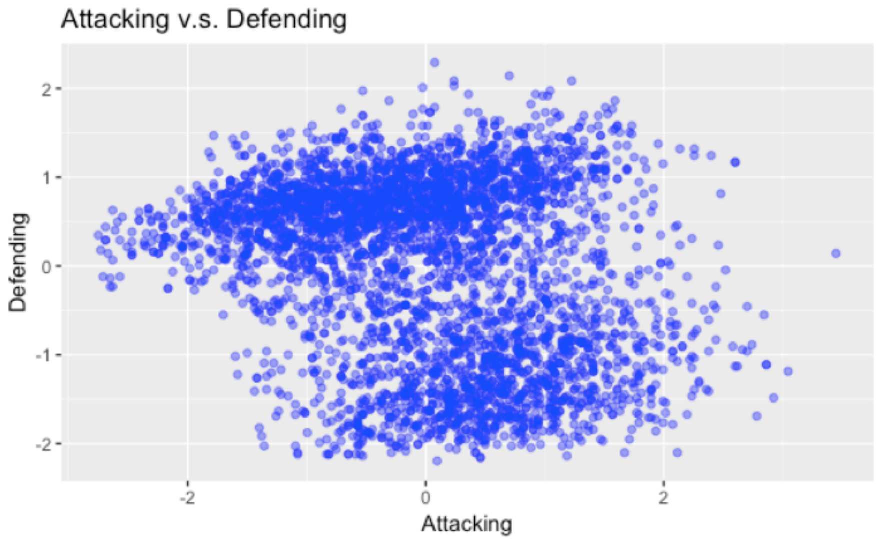
Based on the categorized and scaled data, we drew the boxplots below to show the new distribution of the same variables. In comparison, the variables are balanced after scaling. The observations centered around 0 and have the same standard deviation. However, we can see some observations have higher attacking score than the third quantile, while some have lower movement score than the first quantile. Whether these observations represent outliers requires further examination.



(3) Scatter Plots

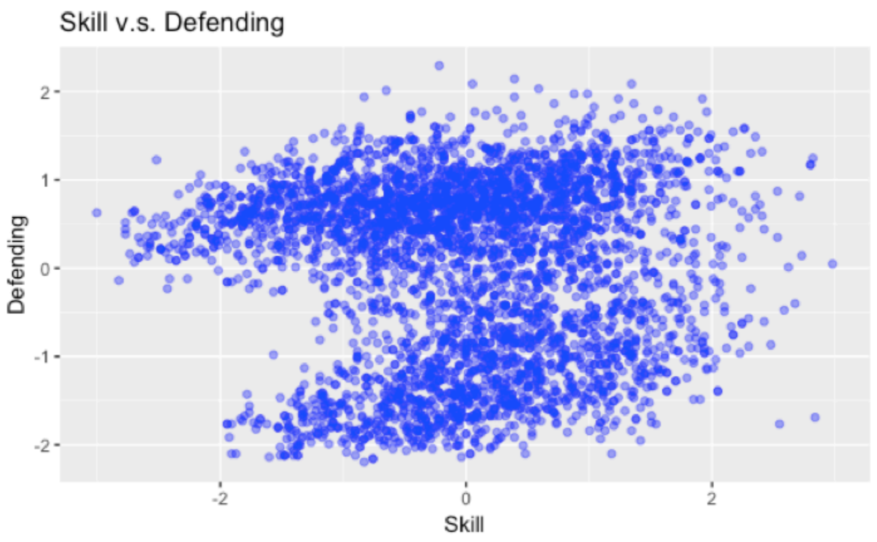
We drew scatter plots of each pair of variables to observe the inherent clusters in the data. To achieve better visualization, we randomly select 4000 observations to plot. Some of the scatterplots do not have a clear pattern of clusters, while some of them show significant pattern of clustering.

1. Attacking and Defending



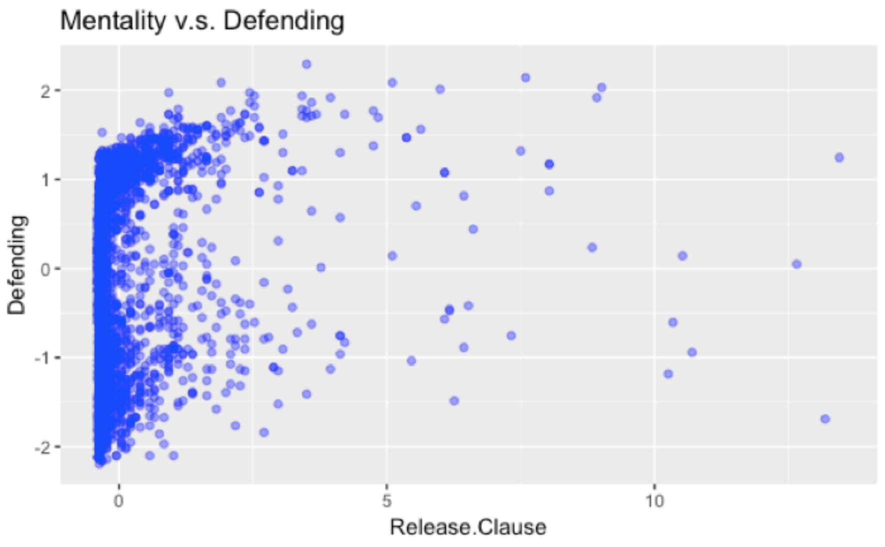
We can discover that there are two clusters of players in this plot. One consists of players that have strong defending ability but are relatively weak in attacking, the other one is the opposite.

2. Skill and Defending



The plot also shows a clear evidence of inherent clusters in the data. There are a bunch of players who are good at defending but relatively less skillful. The other cluster consists of players that are bad at defending but more skillful. One thing worth noticed is that there are more points in the middle of two clusters than that of the first scatter plot.

3. Release.Clause and Defending



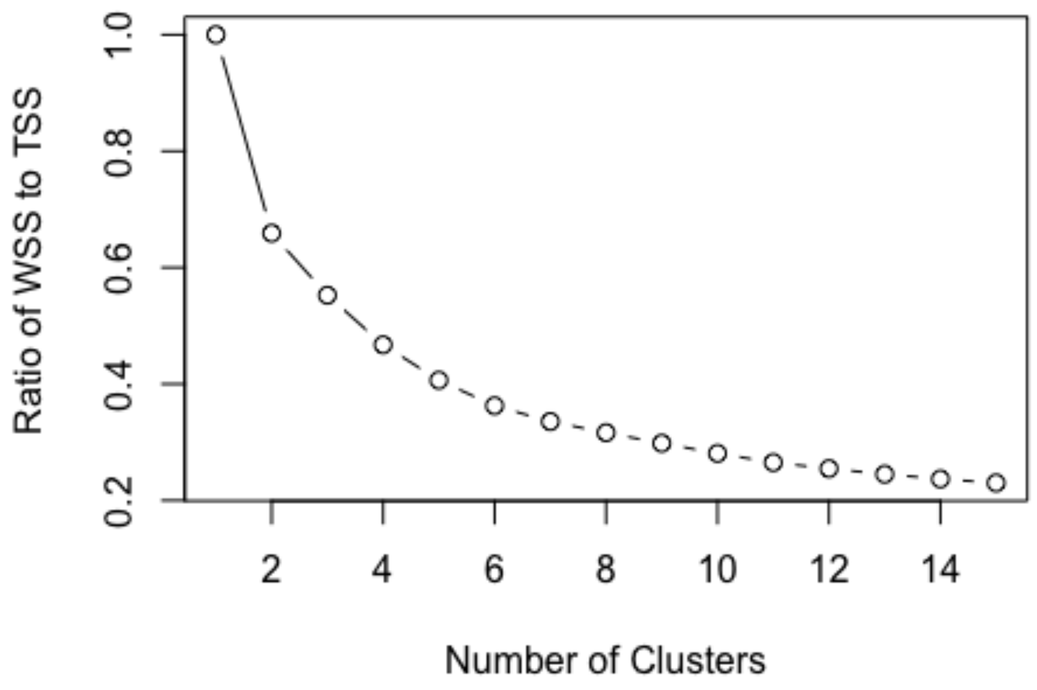
This plot also indicates that players are categorized into two group, one with high defending score and the other with low defending score. However, there is another interesting pattern in this plot. The players are also divided into different groups by their release clause value. If we included more observation, the pattern would become more observable.

Part 1. Discover Clusters

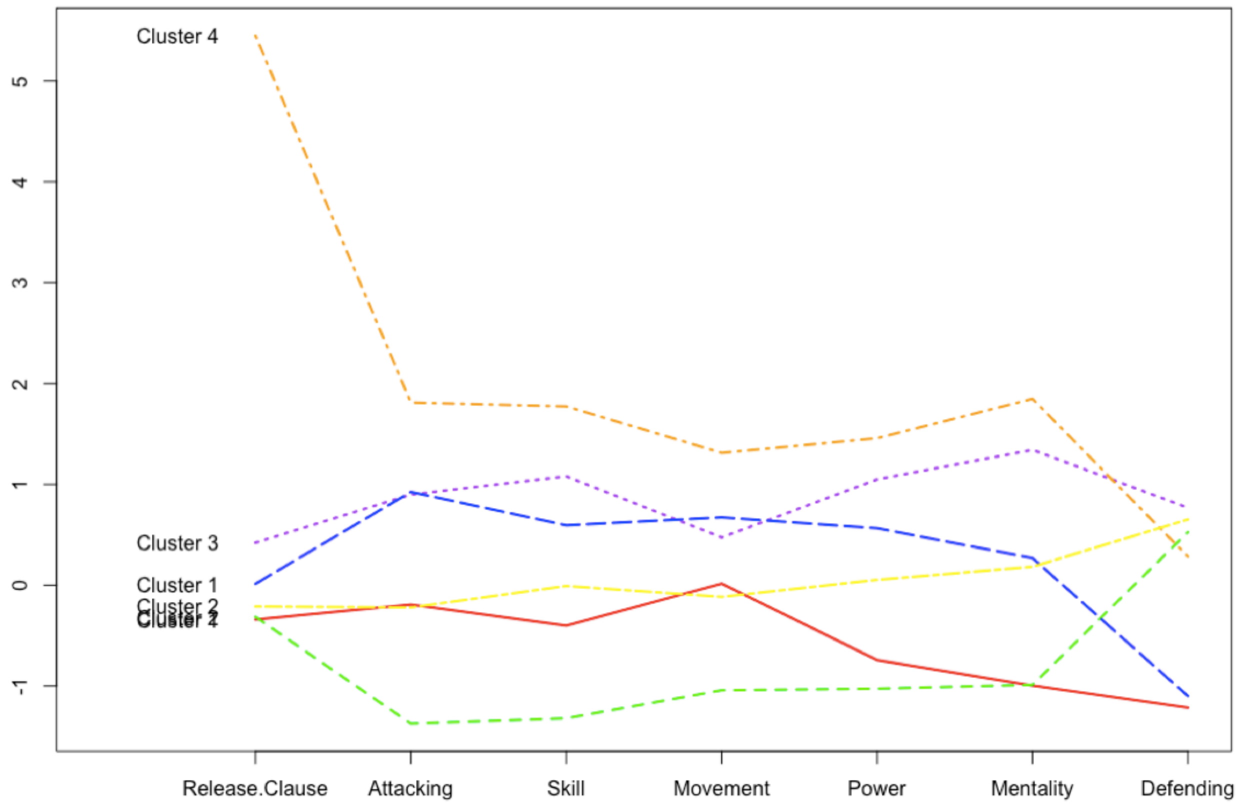
1. K-Means Clustering

(1) Determining Number of Clusters

In order to find the number of clusters inherent in the data, we ran the algorithm multiple times with k from 1 to 15, and below was the acquired scree plot. We determined that the elbow point is 6. The total within cluster sum of squares decreased much slower with addition of another cluster. We chose 6 as the value of k to maintain a balance between the quality of the model and the model complexity.



(2) Profile Analysis

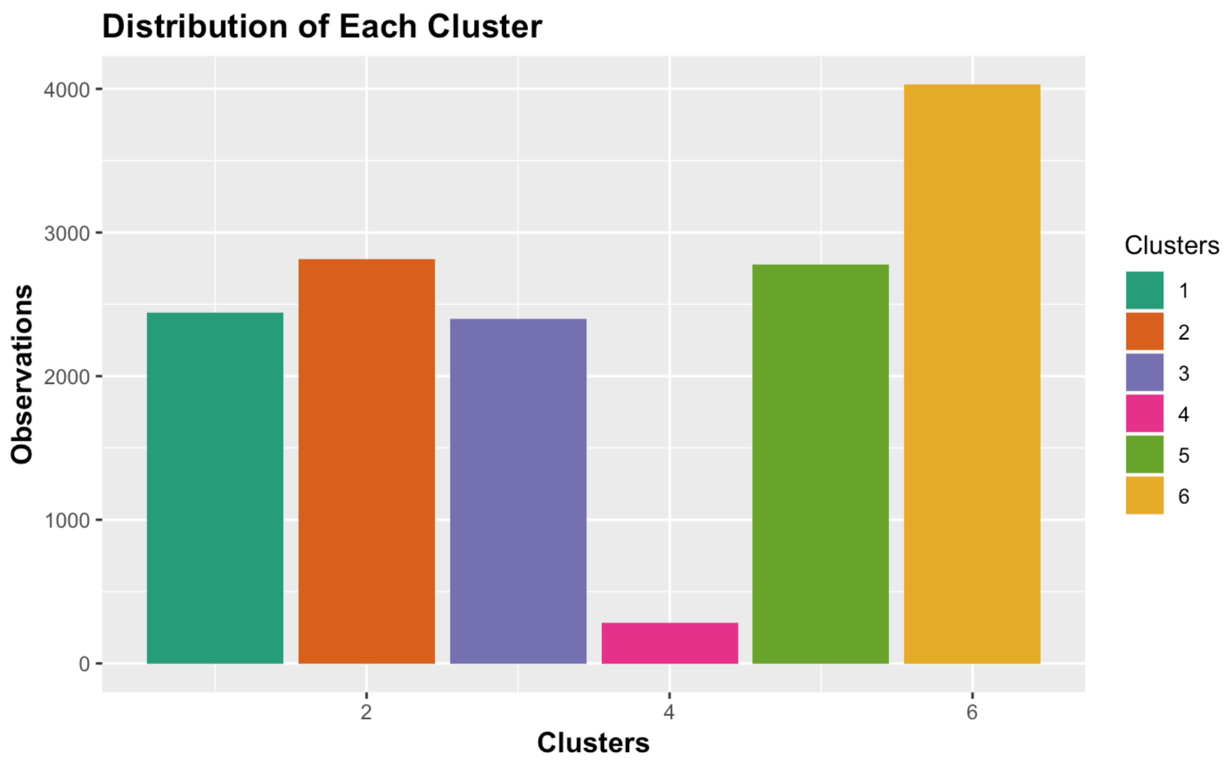


The profile plot shows the difference and similarity of the clusters, and can help us characterize out clusters. From the plot we can discover that the biggest difference is caused by the release clause variable. We inferred that the difference is because of the "star player" effect. Some players are much more famous and are considered more valuable than the non-star players. Defending also creates significant difference. Players have polarized performance on defending.

1. The first cluster is represented by the red line. It is consisted of players that have low release clause value, have average performance on the attacking, skill and movement aspects, and have bad performance on the power, mentality and defending aspects. It has some similarity with the sixth cluster but has different patterns on power, mentality and defending aspects.
2. The second cluster is represented by the green line. It is consisted of players that have low release clause value, have worst performance on the attacking, skills, movement, power and mentality aspects, but are really good at defending.
3. The third cluster is represented by the purple line. It is consisted of players that have average release clause value, rank second on attacking, skill, power and mentality, rank first on defending. It lies right below the "star player" cluster.

4. The fourth cluster is represented by the orange line. It is consisted of those “star” players that have very high release clause value and basically have the best performance on all the rating aspects.
5. The fifth cluster is represented by the blue line. It is consisted of players that have relatively low release clause value, have average performance on the attacking, skills, movement, power and mentality aspects, and are bad at defending.
6. The sixth cluster is represented by the yellow line. It is consisted of players that had low release clause value but have average performance on all the other aspects.

(3) Distribution of Each Cluster

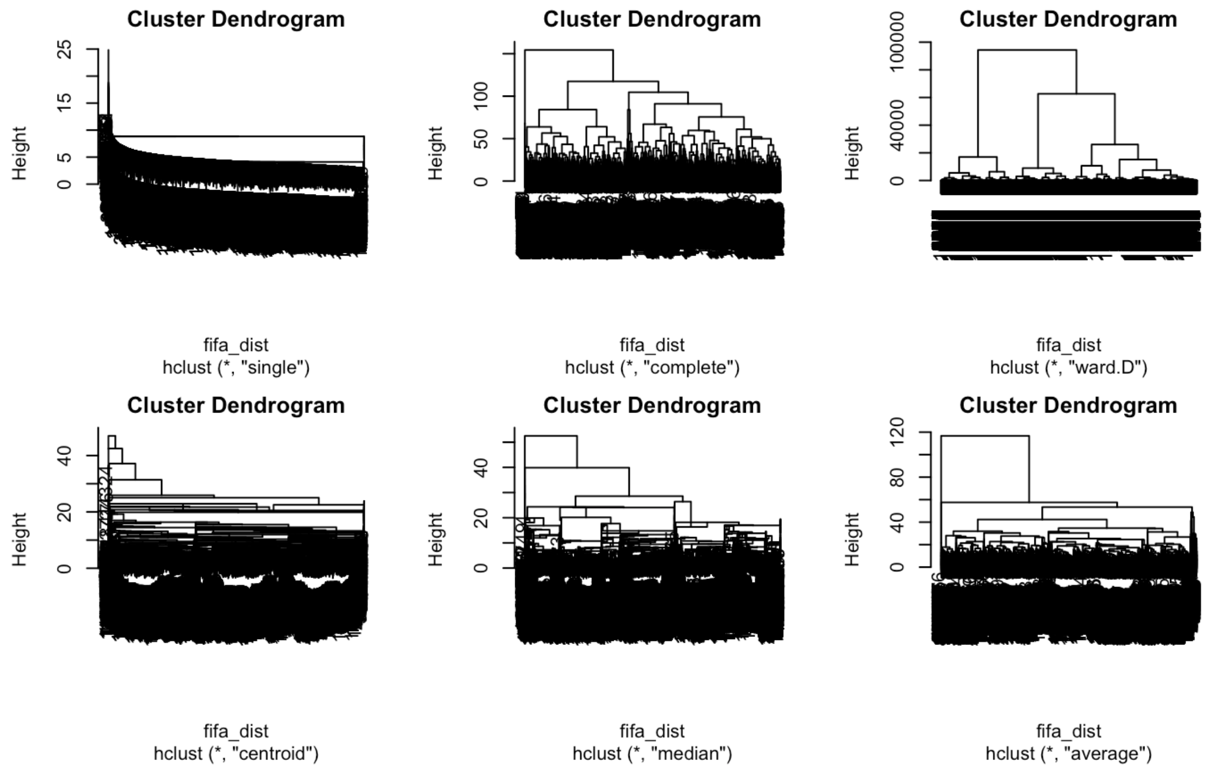


The histogram showed that the distribution of players among different clusters is relatively even, expect the very few observations in cluster 4. As we have discussed, the fourth cluster represents those “star players” who enjoy unparalleled fame and perform better than the other players most of the time. It is consistent with the common sense that this kind of players are rare and valuable.

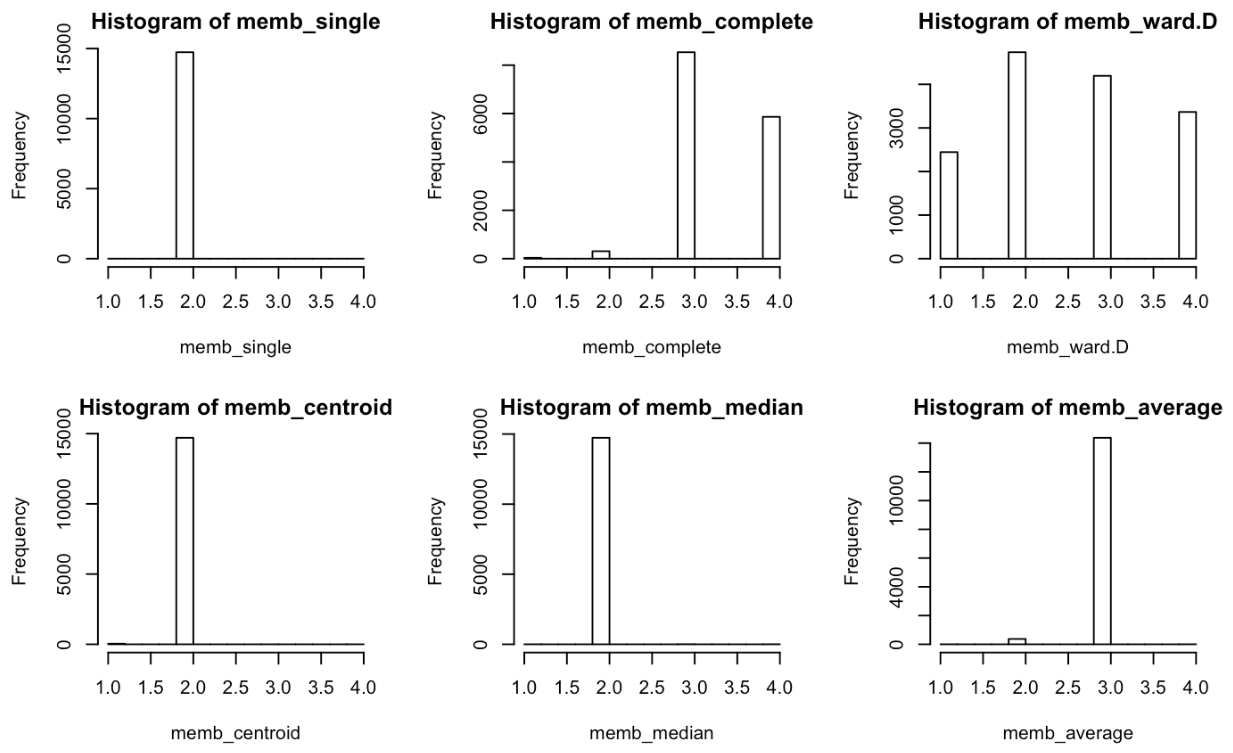
Another interesting pattern we can observe in the histogram is that the number of players in the sixth cluster is higher than the other clusters. According to the profile analysis, the sixth cluster represents players that have average performance on most of the indicators. This demonstrated the fact that plenty of players are in the middle level, they can be the reserve force for new “star players” as long as they focus on improving their overall performance.

2. Hierarchical Clustering

(1) Determining the best method for hierarchical clustering

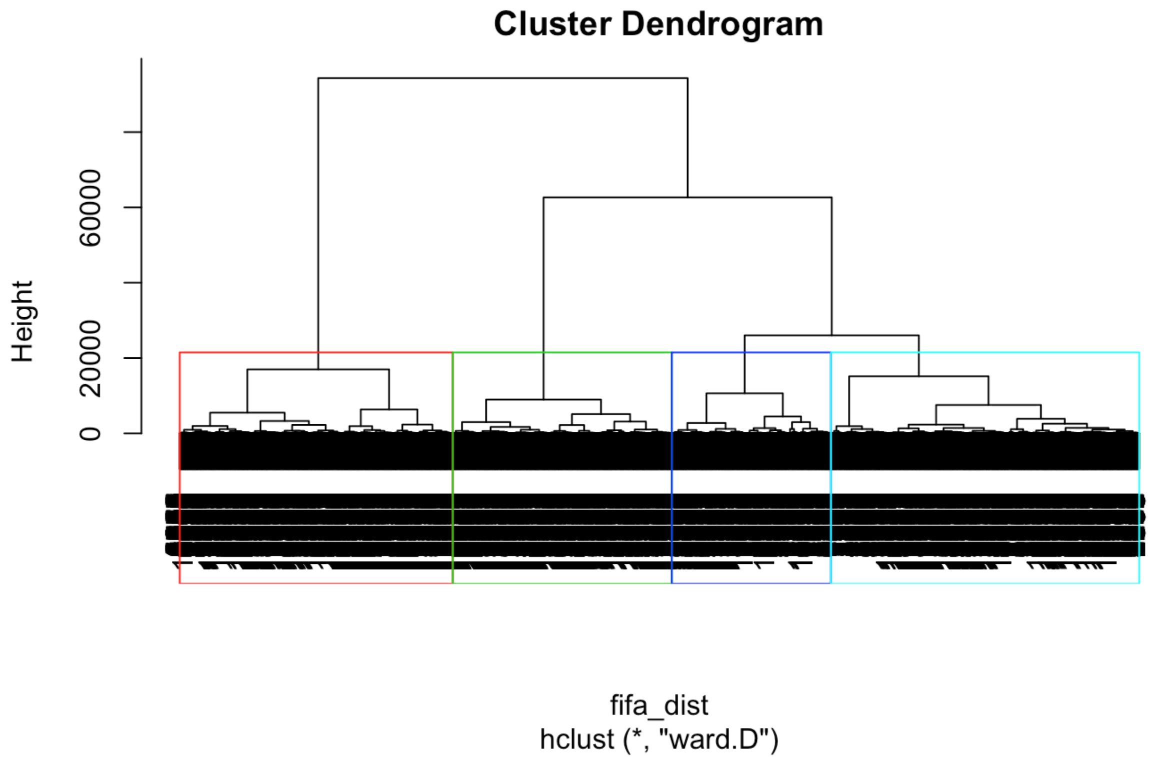


For hierarchical clustering, the most common way of choosing the best method and numbers of clusters is through visual criteria. After computing the distances, we clustered the data based on different methods, and plotted the dendrograms, testing on numbers of k from 4 to 10. It’s obvious from the plots above that the ward.D method generates the most balanced clusters. Therefore, we chose ward.D as the method of hierarchical clustering. We will have a more detailed look at the number of clusters to choose in the next few graphs.



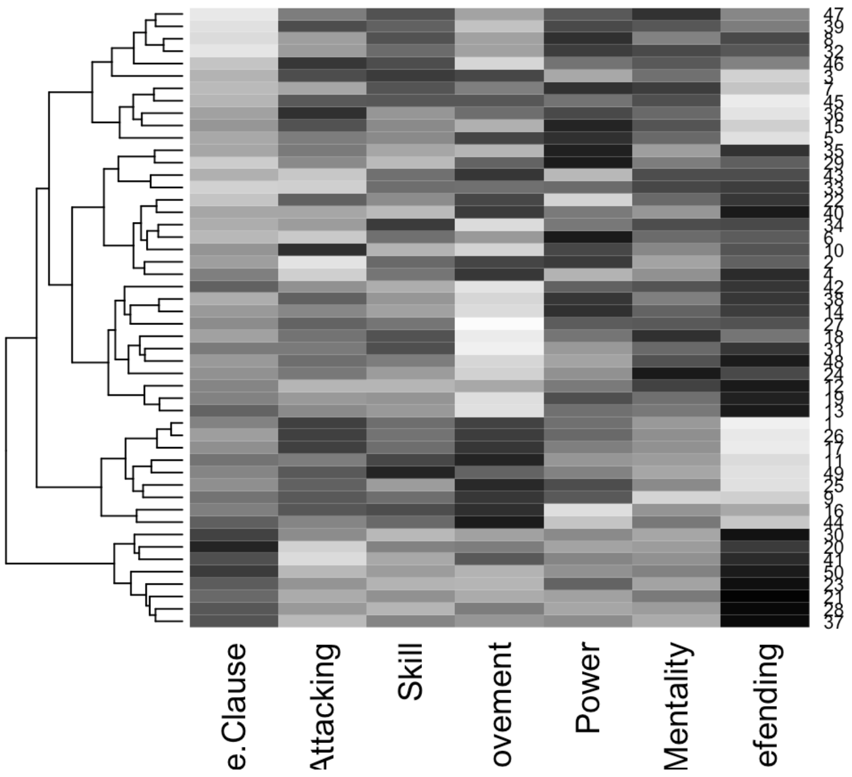
Using the histogram of frequencies for different methods, we double-checked and our conclusion and assured that ward.D is the best method for this dataset as the frequency of different clusters are more balanced than other methods.

(2) Determining number of clusters



When we zoom in and have a deeper look at the dendrogram of the ward.D method, we were more convinced that k=4 is a good choice for generating balanced clusters, which is proved through the rectangles on the dendrogram above.

(3) Heatmap



Heatmap can help us know what are the key factors that determine different clusters. Note that the rows and columns have been sorted in the order specified by the clustering. In the plot above, we set the color to show larger values in darker colors. For example, when we look at the right corner, we can tell that the clusters in the last few lines are dominated by the "defending" scores and "Release clause", differentiating them from other clusters in a higher-level cluster.

(4) Comparing K-means and ward.D hierarchical clustering using Dunn index

Dunn index is an internal measure of clustering validation, with an aim to identify sets of clusters that are compact, with a small variance between members of the cluster, and well separated, where the means of different clusters are sufficiently far apart, as compared to the within cluster variance. For a given assignment of clusters, a higher Dunn index indicates better clustering.

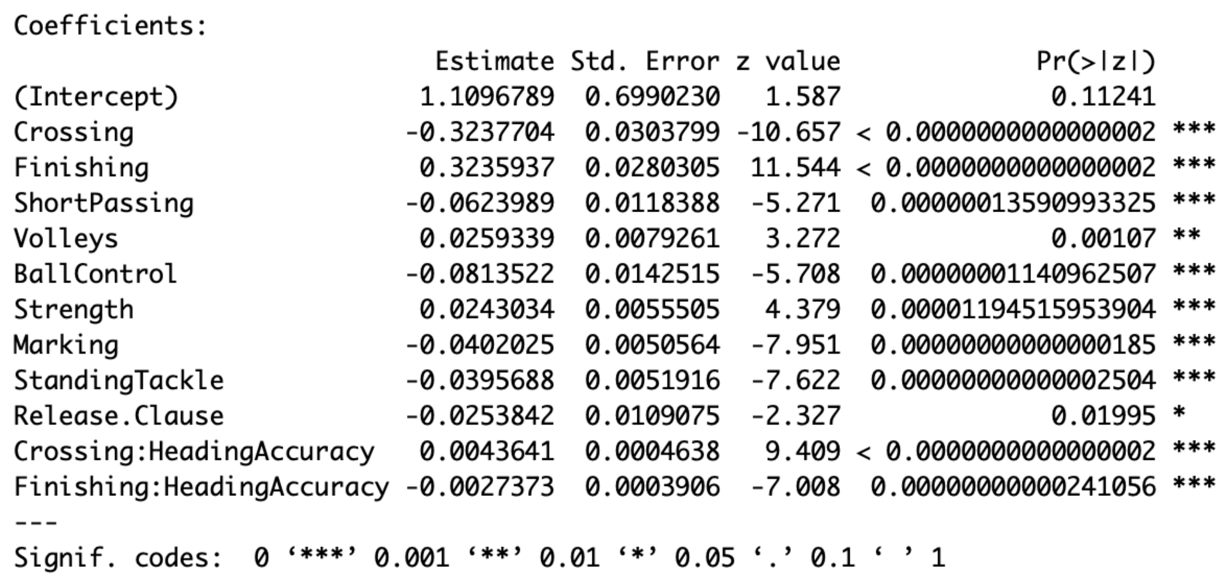
We chose a cluster number of 6 (k=6) to compare the result of two different clustering methods. The result shows that the Dunn index for ward.D hierarchical clustering is a bit higher than K-means clustering, meaning ward.D is a better clustering for this dataset.

```
                     dunn_ward.D
dunn_km               0.0102992382168247
  0.00840682157140454                  1
```

Part2. Revisit Logistic Regression

● Regression Without Clusters

glm(formula = ST ~ Crossing + Finishing + ShortPassing + Volleys + BallControl + Strength + Marking + StandingTackle + Release.Clause + Crossing:HeadingAccuracy + Finishing:HeadingAccuracy, family = "binomial", data = train.df.km)



```
Coefficients:
                            Estimate Std. Error z value             Pr(>|z|)
(Intercept)                1.1096789  0.6990230   1.587              0.11241
Crossing                  -0.3237704  0.0303799 -10.657 < 0.0000000000000002 ***
Finishing                  0.3235937  0.0280305  11.544 < 0.0000000000000002 ***
ShortPassing              -0.0623989  0.0118388  -5.271   0.00000013590993325 ***
Volleys                    0.0259339  0.0079261   3.272              0.00107 **
BallControl               -0.0813522  0.0142515  -5.708   0.00000001140962507 ***
Strength                   0.0243034  0.0055505   4.379   0.00001194515953904 ***
Marking                   -0.0402025  0.0050564  -7.951   0.00000000000000185 ***
StandingTackle            -0.0395688  0.0051916  -7.622   0.00000000000002504 ***
Release.Clause            -0.0253842  0.0109075  -2.327              0.01995 *
Crossing:HeadingAccuracy   0.0043641  0.0004638   9.409 < 0.0000000000000002 ***
Finishing:HeadingAccuracy -0.0027373  0.0003906  -7.008   0.00000000000241056 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Training

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7432  256
##          1  262  895
##
##                Accuracy : 0.9414
##                  95% CI : (0.9363, 0.9462)
##     No Information Rate : 0.8699
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7419
##  Mcnemar's Test P-Value : 0.8261
##
##             Sensitivity : 0.9659
##             Specificity : 0.7776
##          Pos Pred Value : 0.9667
##          Neg Pred Value : 0.7736
##              Prevalence : 0.8699
##          Detection Rate : 0.8402
##    Detection Prevalence : 0.8692
##       Balanced Accuracy : 0.8718
##
##        'Positive' Class : 0
##
```

Validation

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4962  162
##          1  163  611
##
##                Accuracy : 0.9449
##                  95% CI : (0.9388, 0.9506)
##     No Information Rate : 0.8689
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7582
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9682
##             Specificity : 0.7904
##          Pos Pred Value : 0.9684
##          Neg Pred Value : 0.7894
##              Prevalence : 0.8689
##          Detection Rate : 0.8413
##    Detection Prevalence : 0.8688
##       Balanced Accuracy : 0.8793
##                                           
##        'Positive' Class : 0
```

In order to compare the effect of adding clusters into the model, the original best logistic model is included here with its performance in both training and validation dataset. This model includes variables such as Crossing, Finishing and two interaction terms where all the variables are significant. Also, if we look at the accuracy of this model, it has a good performance for both training and validation dataset, with an in-sample accuracy of 0.9414 and an out-of-sample accuracy of 0.9449. The sensitivity of validation dataset is 0.9684, which indicates that around 96% of actual strikers are successfully identified as "Strikers" by the model. On the other side, the specificity of validation dataset is relatively lower, at 0.7904, which indicates that around 79% of actual non-strikers are correctly classified.

• Regression With K-Means Clusters

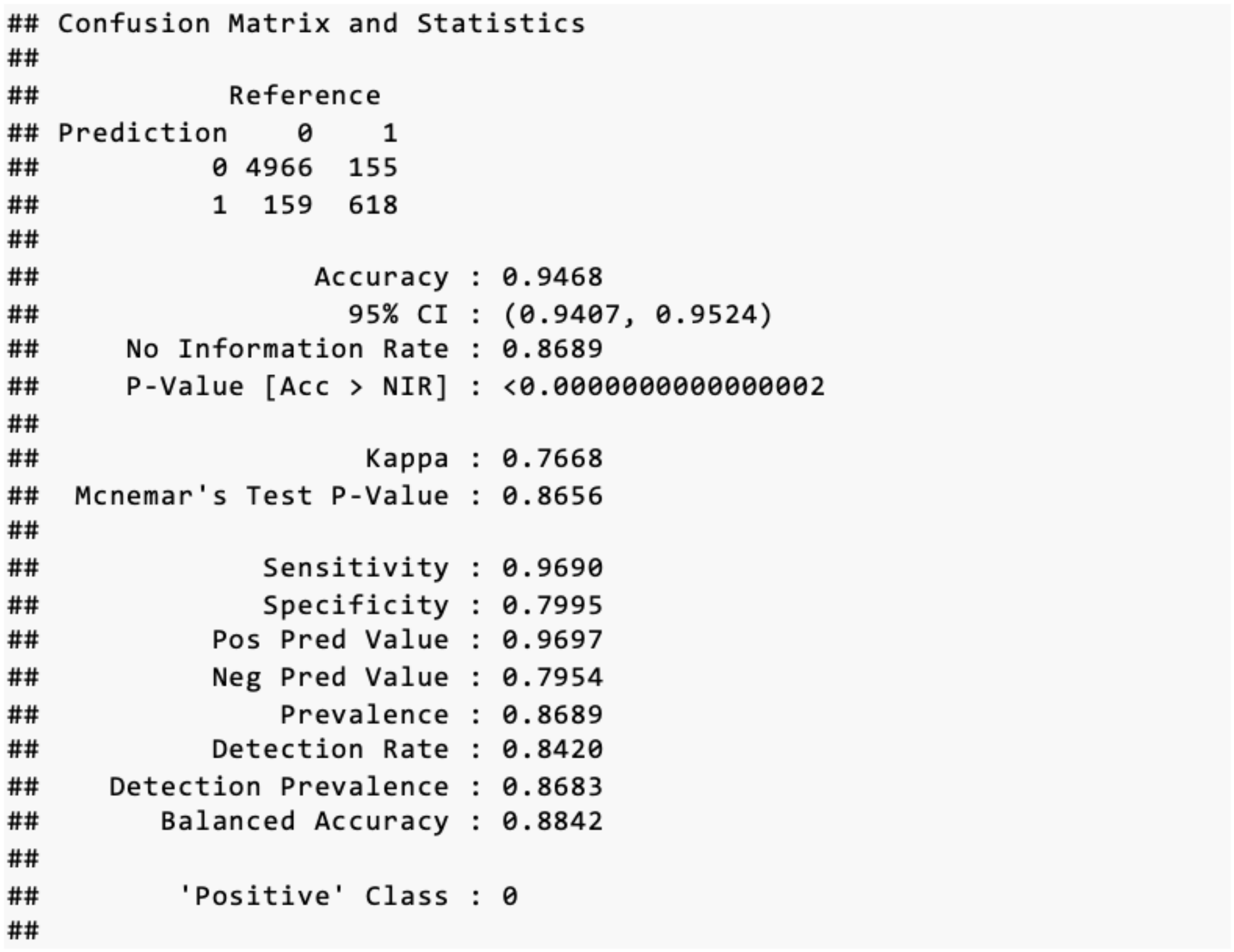
glm(formula = ST ~ Crossing + Finishing + ShortPassing + Volleys + BallControl + Strength + Marking + StandingTackle + Release.Clause + Crossing:HeadingAccuracy + Finishing:HeadingAccuracy + as.factor(fifa_km.cluster), family = "binomial", data = train.df.km)

```
Coefficients:
                              Estimate Std. Error z value              Pr(>|z|)
(Intercept)                  2.0049410  0.9078133   2.209               0.02721 *
Crossing                    -0.3228622  0.0304206 -10.613 < 0.0000000000000002 ***
Finishing                    0.3044647  0.0289091  10.532 < 0.0000000000000002 ***
ShortPassing                -0.0622599  0.0121679  -5.117     0.00000031086942 ***
Volleys                      0.0256559  0.0081822   3.136               0.00172 **
BallControl                 -0.0850371  0.0145935  -5.827     0.00000000564169 ***
Strength                     0.0251237  0.0056819   4.422     0.00000979146801 ***
Marking                     -0.0376240  0.0052148  -7.215     0.00000000000054 ***
StandingTackle              -0.0324836  0.0059503  -5.459     0.00000004783336 ***
Release.Clause              -0.0352365  0.0183290  -1.922               0.05455 .
as.factor(fifa_km.cluster)2 -2.2858038  0.6002787  -3.808               0.00014 ***
as.factor(fifa_km.cluster)3 -0.2210806  0.3384891  -0.653               0.51367
as.factor(fifa_km.cluster)4  0.7311523  0.6429834   1.137               0.25549
as.factor(fifa_km.cluster)5 -0.0046424  0.1832746  -0.025               0.97979
as.factor(fifa_km.cluster)6 -0.4983031  0.3623806  -1.375               0.16911
Crossing:HeadingAccuracy     0.0043158  0.0004644   9.294 < 0.0000000000000002 ***
Finishing:HeadingAccuracy   -0.0026298  0.0003929  -6.694     0.00000000002172 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Training

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7431  255
##          1  263  896
##
##                Accuracy : 0.9414
##                  95% CI : (0.9363, 0.9462)
##     No Information Rate : 0.8699
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7421
##  Mcnemar's Test P-Value : 0.7584
##
##             Sensitivity : 0.9658
##             Specificity : 0.7785
##          Pos Pred Value : 0.9668
##          Neg Pred Value : 0.7731
##              Prevalence : 0.8699
##          Detection Rate : 0.8401
##    Detection Prevalence : 0.8690
##       Balanced Accuracy : 0.8721
##
##        'Positive' Class : 0
##
```

Validation



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4966  155
##          1  159  618
##
##                Accuracy : 0.9468
##                  95% CI : (0.9407, 0.9524)
##     No Information Rate : 0.8689
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7668
##  Mcnemar's Test P-Value : 0.8656
##
##             Sensitivity : 0.9690
##             Specificity : 0.7995
##          Pos Pred Value : 0.9697
##          Neg Pred Value : 0.7954
##              Prevalence : 0.8689
##          Detection Rate : 0.8420
##    Detection Prevalence : 0.8683
##       Balanced Accuracy : 0.8842
##
##        'Positive' Class : 0
##
```

After getting the cluster numbers, we put those numbers back into the dataset and revisited the best model we chose in classification. By adding clusters into the model, the in-sample accuracy does not change at all (remains 0.9414) while the out-of-sample accuracy improved by 0.54%, reaching 0.9468. That is to say, clusters does not help to explain whether a certain player is a striker or not, but would be crucial in predicting whether a certain player is suitable to become a striker or not. As a result , adding clusters generated from k-means clustering into our model will increase out-of-sample model predictability.

Specifically, the sensitivity of validation dataset has been improved from 0.9684 to 0.9690, and the specificity has been improved from 0.7904 to 0.7995. As a result, adding clusters generated from k-means clustering as predictors will help with correctly identifying both actual strikers and actual non-strikers.

In addition, in the model that has clusters included, the coefficient for cluster 2 is significantly different from that of cluster 1 while those for the other clusters are insignificant. The coefficient on cluster 2 (-2.2858038) demonstrates that players in cluster 2 is 89.8% less likely to become strikers, compared with players in cluster 1.

- **Regression With Hierarchy Clusters**

```
glm(formula = ST ~ Crossing + Finishing + ShortPassing + Volleys + BallContro
l + Strength + Marking + StandingTackle + Release.Clause + Crossing:HeadingAc
curacy + Finishing:HeadingAccuracy + as.factor(HierCluster), family = "binomi
al", data = train.df.hr)
```

```
Coefficients:
                           Estimate Std. Error z value             Pr(>|z|)    
(Intercept)               2.8757018  0.8630239   3.332             0.000862 ***
Crossing                 -0.3303731  0.0307638 -10.739 < 0.0000000000000002 ***
Finishing                 0.2967596  0.0286909  10.343 < 0.0000000000000002 ***
ShortPassing             -0.0606751  0.0120925  -5.018     0.00000052321856 ***
Volleys                   0.0235041  0.0080631   2.915             0.003557 ** 
BallControl              -0.0856017  0.0146649  -5.837     0.00000000530927 ***
Strength                  0.0256977  0.0055466   4.633     0.00000360298631 ***
Marking                  -0.0344782  0.0053195  -6.482     0.00000000009080 ***
StandingTackle           -0.0312636  0.0063408  -4.931     0.00000082015892 ***
Release.Clause           -0.0176025  0.0111168  -1.583             0.113327    
as.factor(HierCluster)2  -1.6727995  0.4788044  -3.494             0.000476 ***
as.factor(HierCluster)3  -0.2038703  0.1851639  -1.101             0.270885    
as.factor(HierCluster)4  -3.4823747  0.7871439  -4.424     0.00000968615843 ***
Crossing:HeadingAccuracy  0.0044111  0.0004687   9.411 < 0.0000000000000002 ***
Finishing:HeadingAccuracy -0.0026892  0.0003938  -6.828     0.00000000000861 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Training

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7437  243
##          1  257  908
##
##                Accuracy : 0.9435
##                  95% CI : (0.9385, 0.9482)
##     No Information Rate : 0.8699
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7516
##  Mcnemar's Test P-Value : 0.561
##
##             Sensitivity : 0.9666
##             Specificity : 0.7889
##          Pos Pred Value : 0.9684
##          Neg Pred Value : 0.7794
##              Prevalence : 0.8699
##          Detection Rate : 0.8408
##    Detection Prevalence : 0.8683
##       Balanced Accuracy : 0.8777
##
##        'Positive' Class : 0
##
```

Validation

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4964  160
##          1  161  613
##
##                Accuracy : 0.9456
##                  95% CI : (0.9395, 0.9512)
##     No Information Rate : 0.8689
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.7612
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9686
##             Specificity : 0.7930
##          Pos Pred Value : 0.9688
##          Neg Pred Value : 0.7920
##              Prevalence : 0.8689
##          Detection Rate : 0.8416
##    Detection Prevalence : 0.8688
##       Balanced Accuracy : 0.8808
##
##        'Positive' Class : 0
##
```

Similar as we did for the k-means clustering, we put the hierarchy cluster numbers back into the dataset and revisited the best model we chose for classification. After adding the hierarchy cluster number into the model, both the in-sample accuracy and out-of sample accuracy have been improved. The in-sample accuracy jumped from 0.9414 in the model with no clusters to 0.9435 in the model with hierarchy clusters, and the out-of-sample accuracy increased from 0.9449 to 0.9456. That is to say, adding hierarchy cluster into the model not only helps to explain whether a certain player is a striker or not but also contributes to predicting whether a certain player is suitable to become a striker or not. Therefore, adding clusters generated from hierarchical clustering into our model will also increase out-of-sample model predictability.

The result of the hierarchy also shows that cluster 2 and cluster 4 are significantly different from cluster 1. Specifically, players in cluster 2 is 81.2% less likely to become strikers (with coefficient being -1.6727995) and players in cluster 4 is 96.9% less likely to become strikers (with coefficient being -3.4823747), compared with players in cluster 1.

Additionally, compared with the model that has k-means cluster included, the model with hierarchy clusters performs better in terms of in-sample accuracy, but it is not as good as the k-means model in terms of out-of-sample accuracy. Therefore, we can conclude that the clusters generated from hierarchical clustering are more effective in terms of predicting whether players are strikers compared with the clusters generated from k-means clustering do.

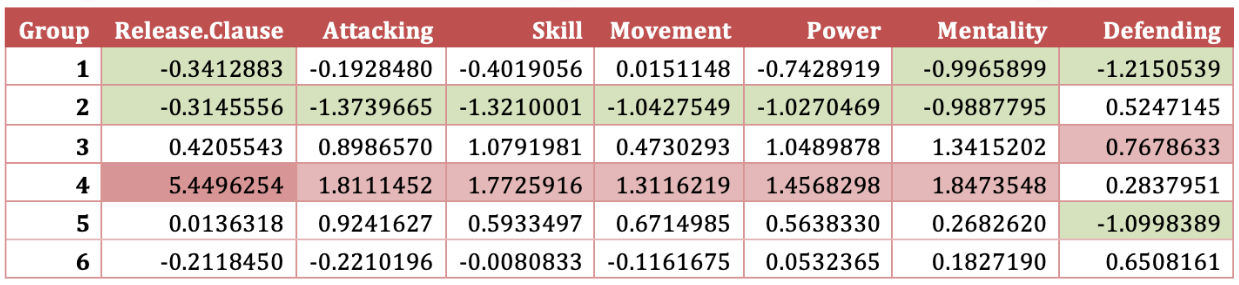
Also, the sensitivity of validation dataset has been improved from 0.9684 to 0.9686, and the specificity has been improved from 0.7904 to 0.7930. As a result, adding clusters generated from k-means hierarchical as predictors will help with correctly identifying both actual strikers and actual non-strikers. However, in the model that has k-means clusters as predictors, the sensitivity reaches 0.9690 and the specificity reaches 0.7995, which also indicate that clusters generated from hierarchical clustering are more effective in terms of predicting whether players are strikers compared with the clusters generated from k-means clustering do.

In conclusion, we find that adding either k-means clusters or hierarchical clusters as predictors into our best logistic regression model will increase model predictability, in terms of classifying potential strikers. Moreover, the clusters generated from hierarchical clustering has a stronger boost in model predictability.

Part3. Characterize Clustering

The membership info tables of clustering are shown from above. We have scaled mean score for each features in each cell. Green represents 'below average', red represents 'above average'.

- ### K-Means Clusters Membership Information



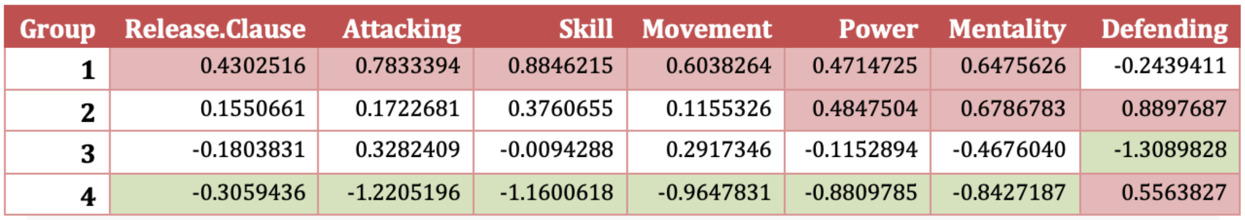
Group	Release.Clause	Attacking	Skill	Movement	Power	Mentality	Defending
1	-0.3412883	-0.1928480	-0.4019056	0.0151148	-0.7428919	-0.9965899	-1.2150539
2	-0.3145556	-1.3739665	-1.3210001	-1.0427549	-1.0270469	-0.9887795	0.5247145
3	0.4205543	0.8986570	1.0791981	0.4730293	1.0489878	1.3415202	0.7678633
4	5.4496254	1.8111452	1.7725916	1.3116219	1.4568298	1.8473548	0.2837951
5	0.0136318	0.9241627	0.5933497	0.6714985	0.5638330	0.2682620	-1.0998389
6	-0.2118450	-0.2210196	-0.0080833	-0.1161675	0.0532365	0.1827190	0.6508161

We found that:

- **Group1: "Below Average"**: Most of the players in group1 perform slightly below the average.

- **Group2 - “Not Ideal”**: Most of the players perform greatly below the average. This is consistent with the result from the logistic regression. Players in group2 have are the least likely to be “Striker” since their poor performance across all features.
- **Group3 - “Defenders”**: players in this group performs on average but pretty good at defending compared to their peers.
- **Group4 - “Strikers”**: Most players in this group perform the best across all groups, especially for the Release Clause and Attacking. This is consistent with the result from the logistic regression. Players in group4 have a higher possibility to be “Sticker” due to their good performance in attacking, skill and etc.
- **Group5 - “Poor defending”**: Players in group5 perform similar to group3 but perform extremely poor in defending.
- **Group6 - “Average”**: Players in this group performs not significant and doesn’t perform better or worse than their peers. Maybe they will fit better with other positions except the ‘Striker’.

● Hierarchical Clusters Membership Information



Group	Release.Clause	Attacking	Skill	Movement	Power	Mentality	Defending
1	0.4302516	0.7833394	0.8846215	0.6038264	0.4714725	0.6475626	-0.2439411
2	0.1550661	0.1722681	0.3760655	0.1155326	0.4847504	0.6786783	0.8897687
3	-0.1803831	0.3282409	-0.0094288	0.2917346	-0.1152894	-0.4676040	-1.3089828
4	-0.3059436	-1.2205196	-1.1600618	-0.9647831	-0.8809785	-0.8427187	0.5563827

We found that:

- **Group1: “Strikers”**: Most of the players in group1 perform slightly below the average.
- **Group2 - “Seeds”**: Players in group2 perform slightly above aEven though we didn’t see there is a fit for the ‘Striker’ position, they could perform well in other positions with good power, mentality and defending.
- **Group3 - “Average”**: players in this group performs on average but not good at defending compared to their peers.
- **Group4 - “Potential Defenders”**: Most players in this group perform poorly across all groups, but really good at defending. This is consistent with the result from the logistic regression. Players in group4 are not ideal for the “Striker” position but they have potentials in defending. Club could take advantage of this point and make them a future defender.