AS Roma & Winning Elevens

Sarah Black, Michael DeGuire, Anthony Meyers, Danny Moncada, Jonathan Watkins

September 29th, 2019

Business Problem:

We have been hired by AS Roma as their resident data scientist to find patterns that our coach can exploit to increase success on the field and decrease failure. Specifically, he is looking for non-obvious patterns because he has learned all of the obvious ones through experience. He has asked us to use associations rules for this analysis and we have been provided data (euro_soccer.sqlite) to conduct our tests.

We have decided to take a multi-stepped approach to our proposed challenge. First, we want to determine what traits are specific to "successful" teams and what traits are specific to "losing" teams. We are going to do this by analyzing the top 4 teams in the top 4 leagues (16 teams total) and by analyzing the bottom 4 teams in the top 4 leagues (16 teams). We have chosen to focus on the top 4 leagues for determining these overall trends because AS Roma is part of Serie A, which is a competitive league. If we had focused on lower-tier leagues, there would be the potential for associations to appear that could not be applied to AS Roma.

After determining what "successful" and "losing" teams do in the top leagues (formations, # of shots, penalties, defense focus, offense focus, etc.), we will shift our attention to what makes great players. Though we would all love to say to simply hire Ronaldo or Messi, we understand that this simple solution is not feasible for AS Roma. However, we may learn insights into what types of players "successful" teams are most looking for. These characteristics may be speed, attacking rating, defense rating, crossing, heading, short pass, etc. After determining what characteristics make up the best players, which in turn make the best teams, we will move onto compiling our conclusions.

With these recommendations, we hope to tell our coach what hidden trends can help AS Roma become a more successful team. We hope to identify key characteristics in players, as well as key insights into team play that can lead AS Roma to victory.

Specifically, we hope to come up with actions that the manager of Roma can take to get AS Roma in the top 4 teams in Serie A to qualify for the UEFA Champions League.

Definition of Success:

The top 4 teams in each league go on to the Championship which is tied to financial reward and prestige; finishing each season in the top 4 teams is what our team is defining as success. While it is hard to guarantee a top 4 finish, we are aiming to greatly increase the likelihood of Roma finishing there.

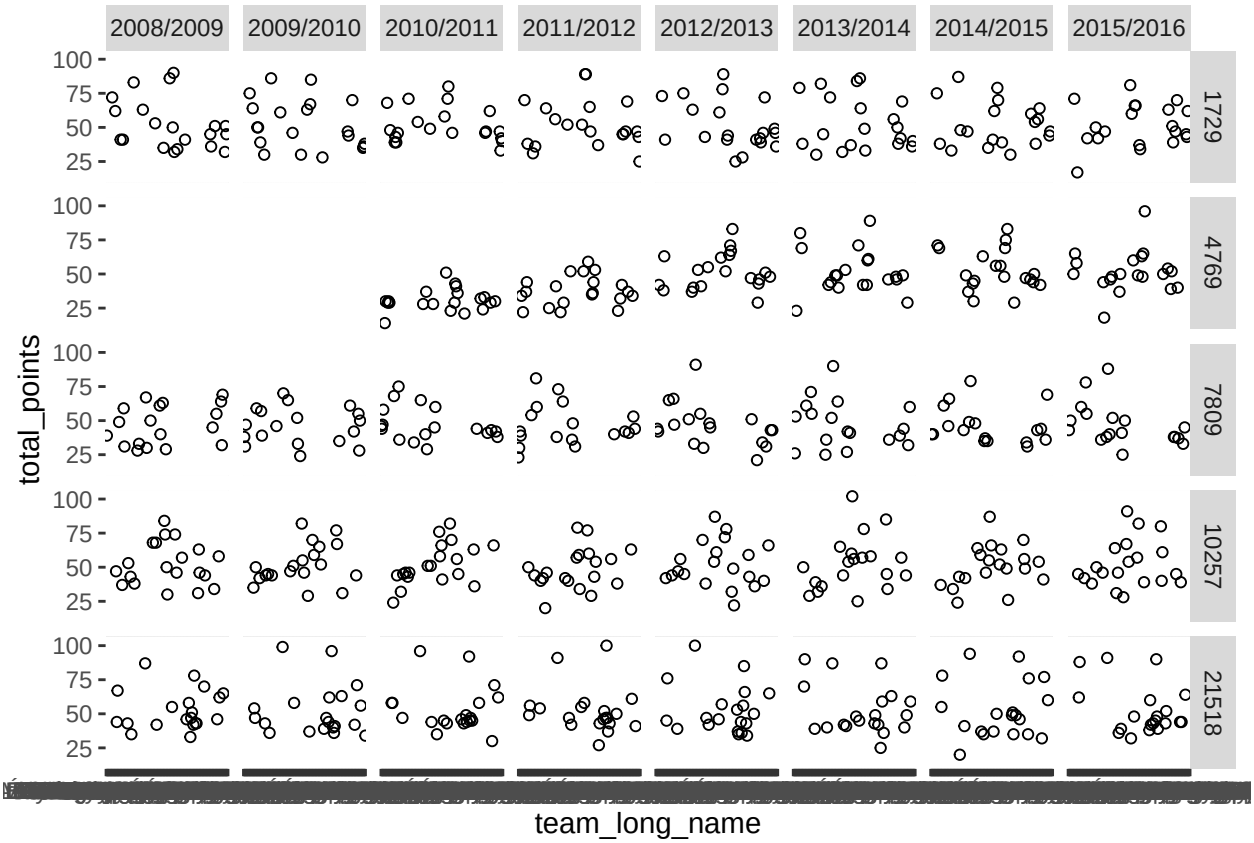
1. K Means Clustering for Team Characteristics

What types of teams are Roma likely to face?

Data Preparation - K Means Clustering

1. Create help functions to unnest XML columns in data
2. Unnested data and created aggregate views of the matches data

```
## [1] "Country"           "League"            "Match"
## [4] "Player"            "Player_Attributes" "Team"
## [7] "Team_Attributes"   "sqlite_sequence"
```




```
## # A tibble: 12,895 x 7
## # Groups:   team [161]
##     team subtype         type  player1 goals  pens    og
##    <dbl> <fct>           <fct>   <dbl> <int> <int> <int>
##  1     0 shot            goal        0     0     0     0
##  2     0 <NA>            goal        0     0     0     0
##  3  4087 deflected       goal   192319     1     0     0
##  4  4087 direct_freekick goal    41194     1     0     0
##  5  4087 direct_freekick goal    74169     3     0     0
##  6  4087 direct_freekick goal   187669     1     0     0
##  7  4087 distance        goal    41194     1     0     0
##  8  4087 distance        goal   210466     1     0     0
##  9  4087 header          goal    40558     1     0     0
## 10  4087 header          goal    46708     1     0     0
## # ... with 12,885 more rows
```

```
full$home_team_api_id=as.integer(full$home_team_api_id)
agg_goals$team = as.integer(agg_goals$team)
agg_shotsoff$team = as.integer(agg_shotsoff$team)
agg_shotson$team = as.integer(agg_shotson$team)
agg_foulcommit$team = as.integer(agg_foulcommit$team)
agg_card$team = as.integer(agg_card$team)
agg_crosses$team = as.integer(agg_crosses$team)
agg_corner$team = as.integer(agg_corner$team)
agg_possession$home_team_api_id = as.integer(agg_possession$home_team_api_id)
```

```
full = full %>% left_join(agg_shotsoff %>% group_by(season, team) %>%
                            summarize(shotsoff = sum(shotoff)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_shotson %>% group_by(season, team) %>%
                            summarize(shotson = sum(shoton)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_foulcommit %>% group_by(season, team) %>%
                            summarize(foulscommitted = sum(foulcommits)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_card %>% group_by(season, team) %>%
                            summarize(yellows = sum(yellows), reds = sum(reds)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_crosses %>% group_by(season, team) %>%
                            summarize(crosses = sum(cross)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_corner %>% group_by(season, team) %>%
                            summarize(corners = sum(corners)),
                          by = c("season"="season", "home_team_api_id"="team"))

full = full %>% left_join(agg_possession,
                          by = c("season"="season", "home_team_api_id"="home_team_api_id"))


#=====================================================
#get top 4 teams by league by season

full %>% filter(home_team_api_id == roma_record$team_api_id)
```

```
## # A tibble: 8 x 23
## # Groups:   season, league_id [8]
##   season league_id home_team_api_id home_points  gf.x  ga.x away_points
##   <chr>      <int>            <int>       <dbl> <int> <int>       <dbl>
## 1 2008/~     10257             8686          43    35    23          20
## 2 2009/~     10257             8686          46    36    17          31
## 3 2010/~     10257             8686          38    31    18          25
## 4 2011/~     10257             8686          35    38    20          21
## 5 2012/~     10257             8686          35    40    24          24
## 6 2013/~     10257             8686          48    44     9          37
## 7 2014/~     10257             8686          37    31    14          33
## 8 2015/~     10257             8686          44    44    17          36
## # ... with 16 more variables: gf.y <int>, ga.y <int>, total_points <dbl>,
## #   gf <int>, ga <int>, shotsoff <int>, shotson <int>,
## #   foulscommitted <int>, yellows <int>, reds <int>, crosses <int>,
## #   corners <int>, home_firsthalf_pos <dbl>, home_secondhalf_pos <dbl>,
## #   away_firsthalf_pos <dbl>, away_secondhalf_pos <dbl>
```

```r
filter_top4 = full %>% group_by(season, league_id) %>%
  arrange(season, league_id, desc(total_points)) %>%
  mutate(rank = row_number(desc(total_points))) %>%
  filter(rank <= 4) %>%
  select(season, league_id, home_team_api_id)

filter_top4_pt2 = full %>% group_by(season, league_id) %>%
  arrange(season, league_id, desc(total_points)) %>%
  mutate(rank = row_number(desc(total_points))) %>%
  filter(rank <= 4) %>%
  select(season, league_id, home_team_api_id)

full_top4 = full %>% left_join(filter_top4,
                               by = c("season"="season", "league_id"="league_id",
                                      "home_team_api_id"="home_team_api_id"))

full_top4 = full_top4 %>% ungroup() %>%
  select(home_gf = gf.x, home_ga = ga.x, away_gf = gf.y, away_ga = ga.y, shotsoff, shotson,
         foulscommitted, yellows, reds, corners, crosses, home_firsthalf_pos,
         home_secondhalf_pos, away_firsthalf_pos, away_secondhalf_pos)

#================================================
#perform kmeans to generate clusters

#normalize the data for k-means
normalize <- function(x){
  return ((x - min(x))/(max(x) - min(x)))}


data_normalized = mutate(full_top4 %>% drop_na(),
                         home_gf = normalize(home_gf),
                         home_ga = normalize(home_ga),
                         away_gf = normalize(away_gf),
                         away_ga = normalize(away_ga),
                         shotsoff = normalize(shotsoff),
                         shotson = normalize(shotson),
                         foulscommitted = normalize(foulscommitted),
                         yellows = normalize(yellows),
                         reds = normalize(reds),
                         corners = normalize(corners),
                         crosses = normalize(crosses),
                         home_firsthalf_pos = normalize(home_firsthalf_pos),
                         home_secondhalf_pos = normalize(home_secondhalf_pos),
                         away_firsthalf_pos = normalize(away_firsthalf_pos),
                         away_secondhalf_pos = normalize(away_secondhalf_pos))

# View(data_normalized)


# kmeans

library(stats)
```

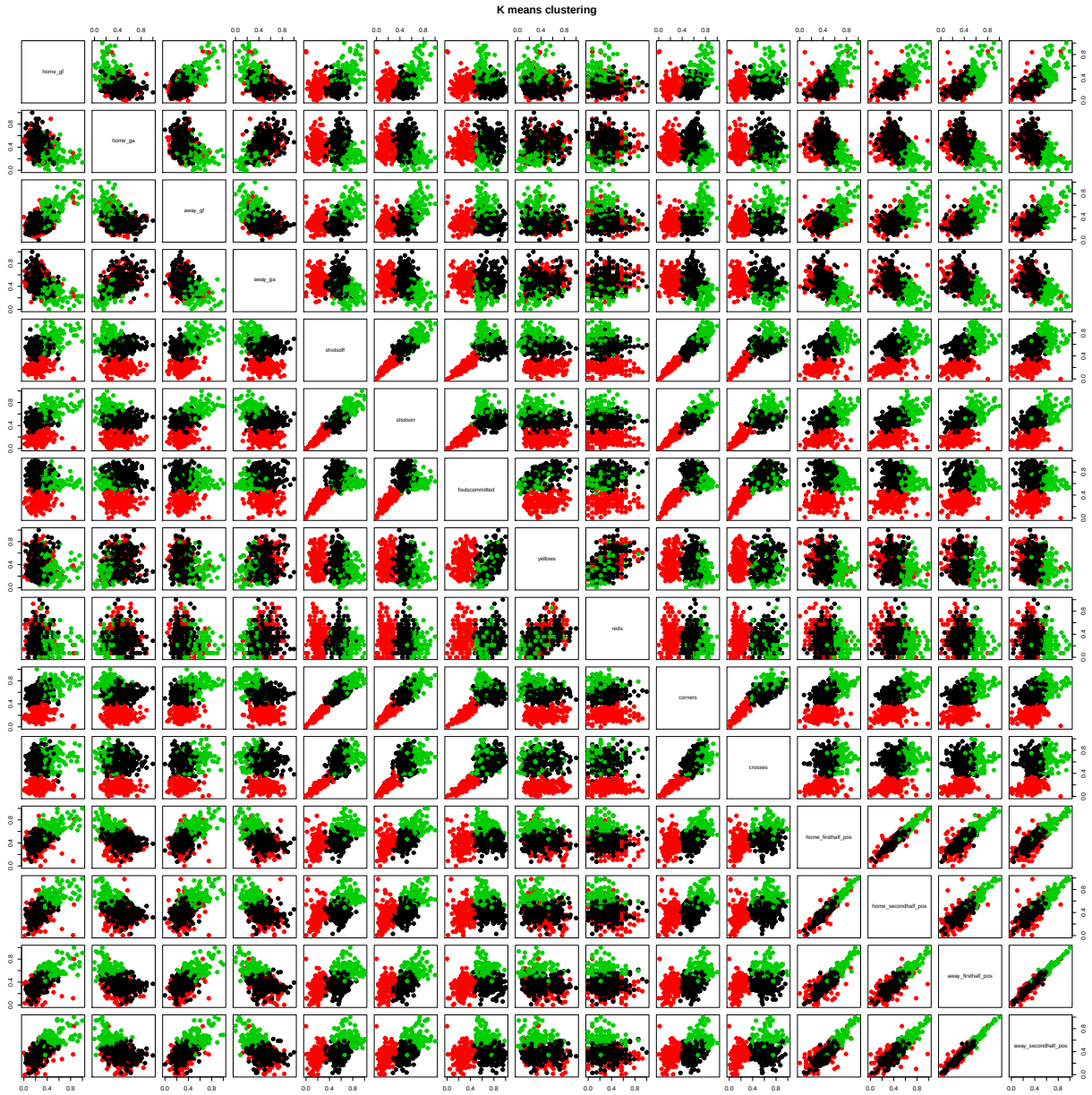
```r
kcluster <- kmeans(data_normalized, 3)
kcluster$size
```

```
## [1] 258 174 115
```

```r
kcluster$centers
```

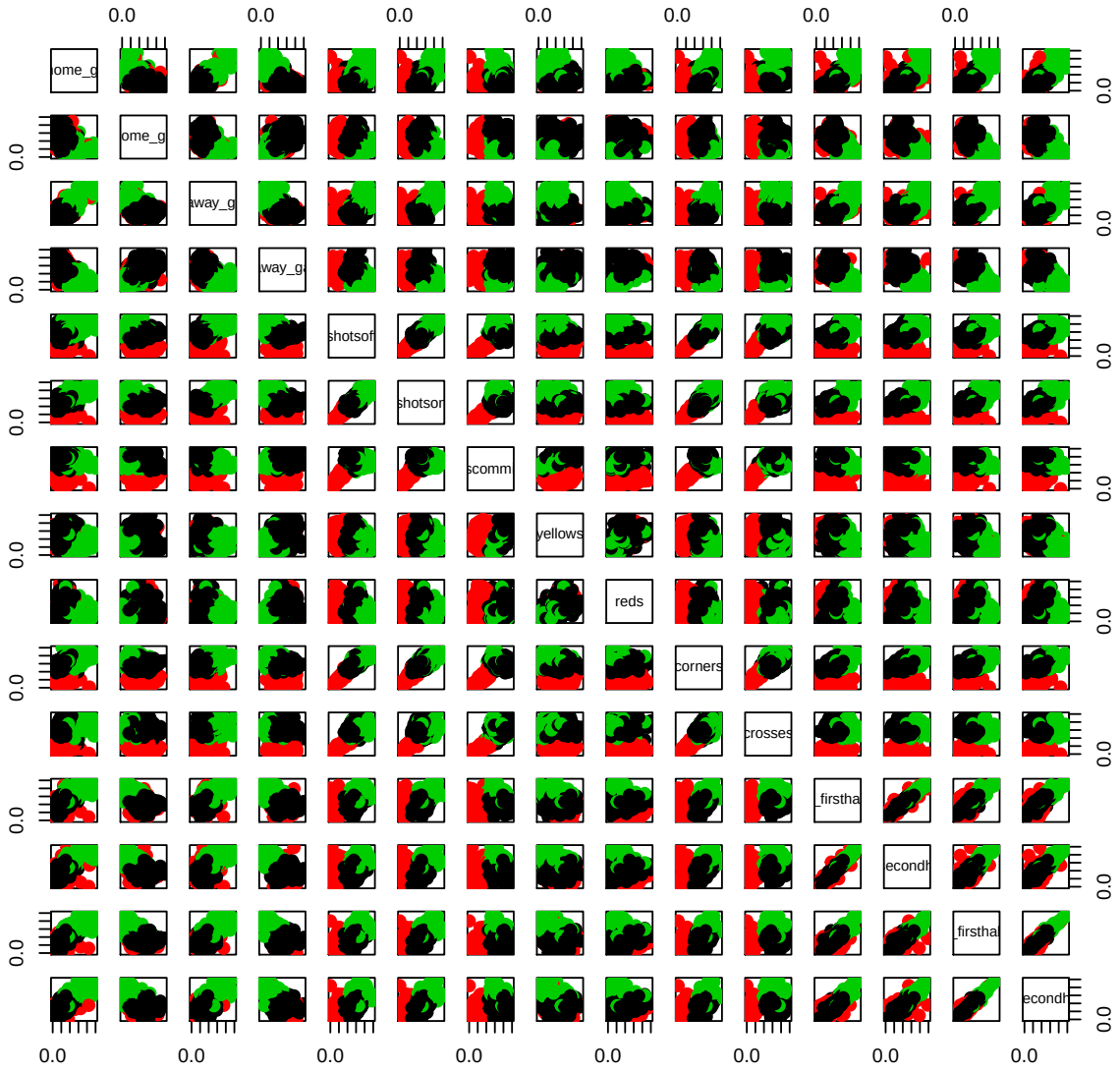
```
##     home_gf   home_ga   away_gf   away_ga  shotsoff   shotson
## 1 0.2464009 0.4696208 0.2923934 0.5008075 0.5719958 0.4956712
## 2 0.2599891 0.4280833 0.2927443 0.4904215 0.2055081 0.1753948
## 3 0.4924776 0.2622797 0.5295290 0.2985507 0.7160750 0.6678180
##   foulscommitted   yellows      reds   corners   crosses
## 1      0.7181751 0.4079085 0.3095238 0.5783226 0.6057134
## 2      0.2875792 0.4439655 0.3665846 0.2085791 0.1611785
## 3      0.6511203 0.2985786 0.2540373 0.7308352 0.6728592
##   home_firsthalf_pos home_secondhalf_pos away_firsthalf_pos
## 1          0.4321140           0.4018861          0.3607998
## 2          0.3789848           0.3464961          0.3163705
## 3          0.6734655           0.6628942          0.6196981
##   away_secondhalf_pos
## 1           0.3608122
## 2           0.3219167
## 3           0.6236912
```

```r
# kcluster$cluster # Exclude from analysis
plot(data_normalized, col = (kcluster$cluster), main = "K means clustering",
     pch = 20, cex = 2)
```



Same plot as above at different plot dim

```
plot(data_normalized, col = (kcluster$cluster), main = "K means clustering",
     pch = 20, cex = 2)
```

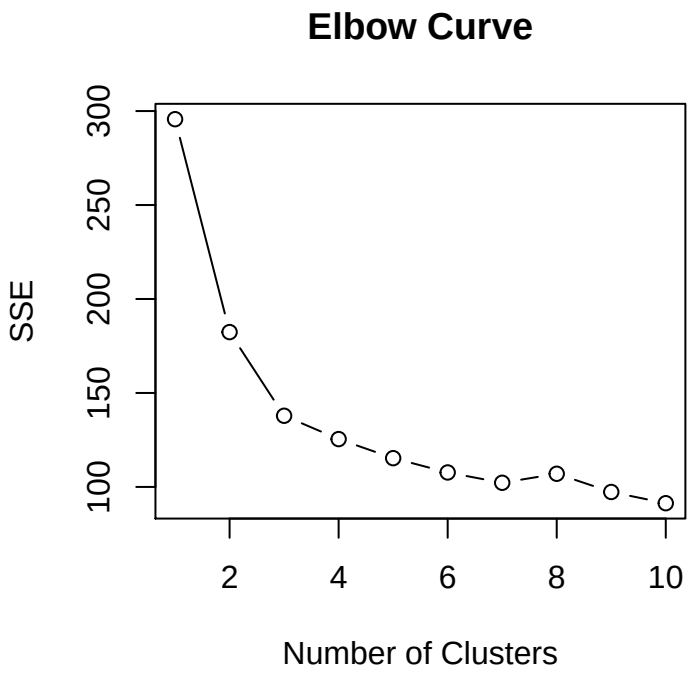



```
SSE_curve <- c()
for (n in 1:10) {
  kcluster <- kmeans(data_normalized, n)
  print(kcluster$withinss)
  sse <- sum(kcluster$withinss)
  SSE_curve[n] <- sse
}
```

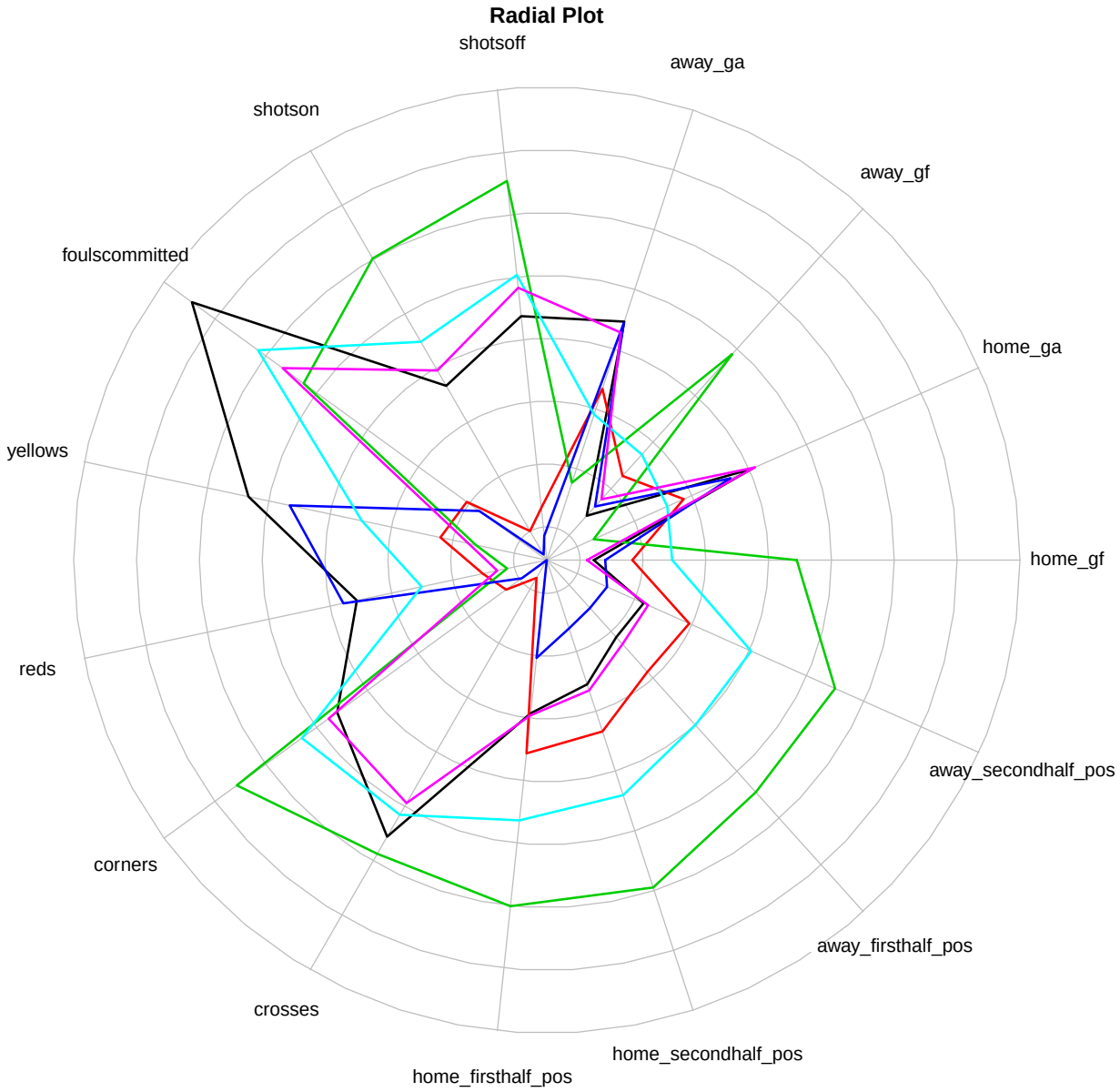
```
## [1] 295.6694
## [1] 135.50007  46.88654
## [1] 58.12982 33.44438 46.29621
## [1] 31.18458 46.59390 21.67142 26.01172
## [1] 18.05439 21.67142 30.32599 19.25203 26.01172
```

```
## [1] 20.078239 19.252034 21.518995  9.537484 18.054388 19.256890
## [1] 17.793638 16.157470 18.384332 12.807462  6.414322 12.603971 18.054388
## [1]  7.450934 46.278804  6.525738  3.742822 10.626659  6.675712 15.272484
## [8] 10.432203
## [1] 21.005520 18.644264  3.734865  3.024962  7.710640  5.210907  9.614152
## [8]  9.009550 19.357055
##  [1] 10.186385  5.626070  8.760640 14.163960  8.081620  8.258060 10.801777
##  [8]  6.482272  6.531667 12.451948
```

```r
plot(1:10, SSE_curve, type="b", xlab="Number of Clusters", ylab="SSE", main = "Elbow Curve")
```




```r
kcluster <- kmeans(data_normalized, 6)
cluster.stats = kcluster$centers

radial.plot(cluster.stats, rp.type = "p", lwd = 2, labels = colnames(cluster.stats), show.grid.labels=0
```




```
#==================================
#merge clusters with the data
full_top4 %>% drop_na() #this is the base data
```

```
## # A tibble: 547 x 15
##    home_gf home_ga away_gf away_ga shotsoff shotson foulscommitted yellows
##      <int>   <int>   <int>   <int>    <int>   <int>          <int>   <int>
##  1      33      12      35      12      332     203            430      50
##  2      40      18      18      32      234     156            471      51
##  3      26      29      12      28      234     124            443      62
##  4      21      25      13      29      203     120            383      69
##  5      17      18      17      27      289     145            463      61
##  6      17      20      11      37      227     123            432      55
```

```
##  7      21      21      20      32      217     127             512      61
##  8      21      10      24      35      223     148             434      56
##  9      41      13      36      14      324     164             440      50
## 10      23      22      19      23      211     142             484      68
## # ... with 537 more rows, and 7 more variables: reds <int>, corners <int>,
## #   crosses <int>, home_firsthalf_pos <dbl>, home_secondhalf_pos <dbl>,
## #   away_firsthalf_pos <dbl>, away_secondhalf_pos <dbl>
```

```
##kcluster$cluster ## hide the clusters

full_top4wTeam = full %>% left_join(filter_top4, by = c("season"="season",
                                                        "league_id"="league_id",
                                                        "home_team_api_id"="home_team_api_id"))

full_top4wTeam = full_top4wTeam %>% ungroup() %>% select(home_team_api_id,
                                                         home_gf = gf.x,
                                                         home_ga = ga.x,
                                                         away_gf = gf.y,
                                                         away_ga = ga.y,
                                                         shotsoff, shotson,
                                                         foulscommitted, yellows,
                                                 reds, corners, crosses,
                                                 home_firsthalf_pos,
                                                 home_secondhalf_pos,
                                                 away_firsthalf_pos,
                                                 away_secondhalf_pos)

#the above is the same data but added team_id so we can track

clustered_full_top4wTeam = full_top4wTeam %>% drop_na()
clustered_full_top4wTeam["cluster"] = kcluster$cluster

#get Romas index' from the cluster table so we can pop it out of the cluster grid
roma_rows=which(clustered_full_top4wTeam$home_team_api_id == roma_record$team_api_id)

data_normalized[roma_rows,]
```

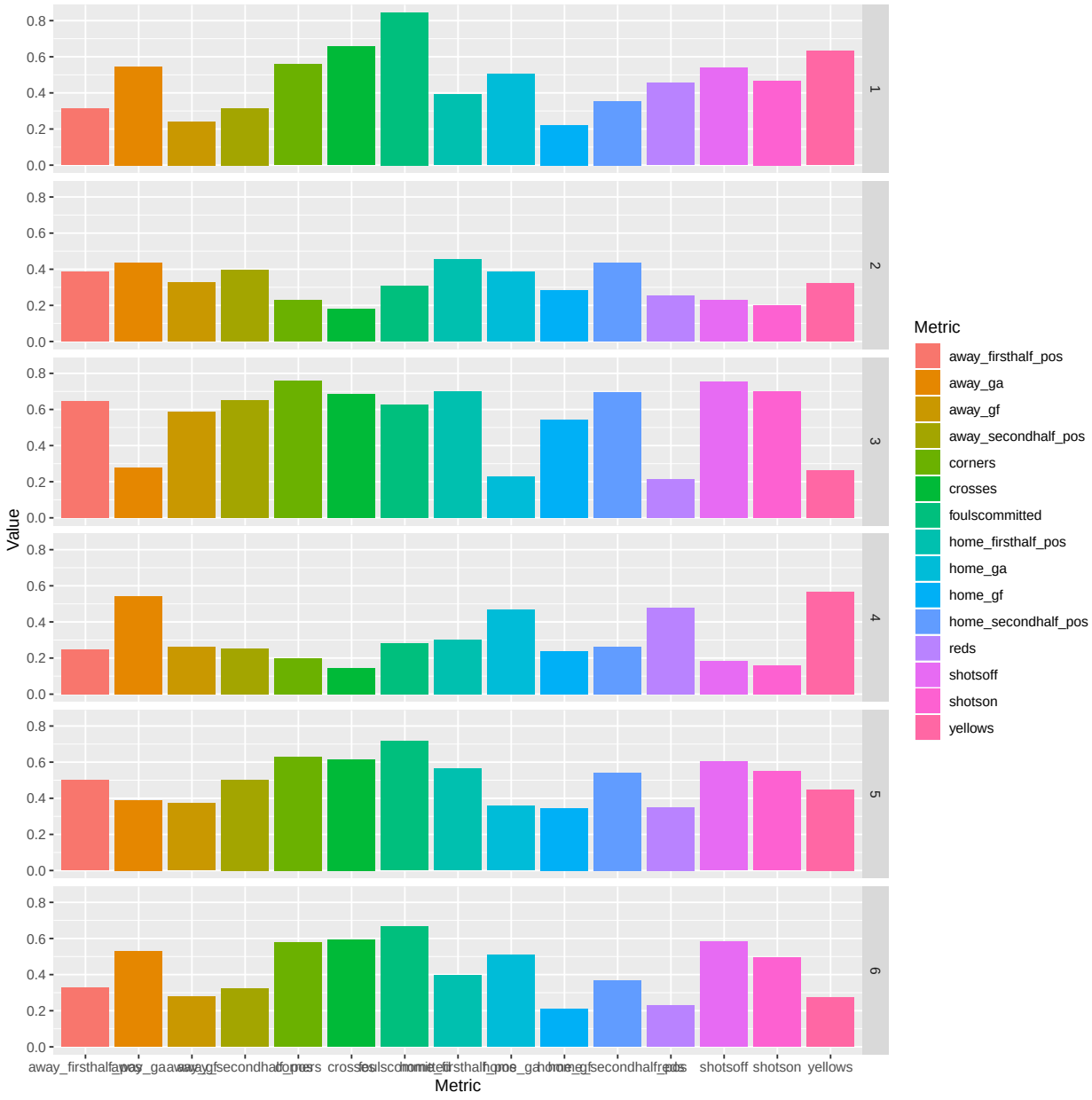
```
## # A tibble: 6 x 15
##   home_gf home_ga away_gf away_ga shotsoff shotson foulscommitted yellows
##     <dbl>   <dbl>   <dbl>   <dbl>    <dbl>   <dbl>          <dbl>   <dbl>
## 1   0.397   0.459   0.5     0.646    0.578   0.561          0.543   0.462
## 2   0.413   0.297   0.521   0.333    0.679   0.556          0.554   0.452
## 3   0.333   0.324   0.479   0.562    0.713   0.561          0.712   0.462
## 4   0.444   0.378   0.333   0.521    0.572   0.444          0.496   0.365
## 5   0.333   0.216   0.375   0.208    0.691   0.514          0.777   0.683
## 6   0.540   0.297   0.708   0.354    0.758   0.654          0.790   0.452
## # ... with 7 more variables: reds <dbl>, corners <dbl>, crosses <dbl>,
## #   home_firsthalf_pos <dbl>, home_secondhalf_pos <dbl>,
## #   away_firsthalf_pos <dbl>, away_secondhalf_pos <dbl>
```

```
#get the last possible year for roma which is last entry in roma_rows
data_normalized[tail(roma_rows, n=1),]
```

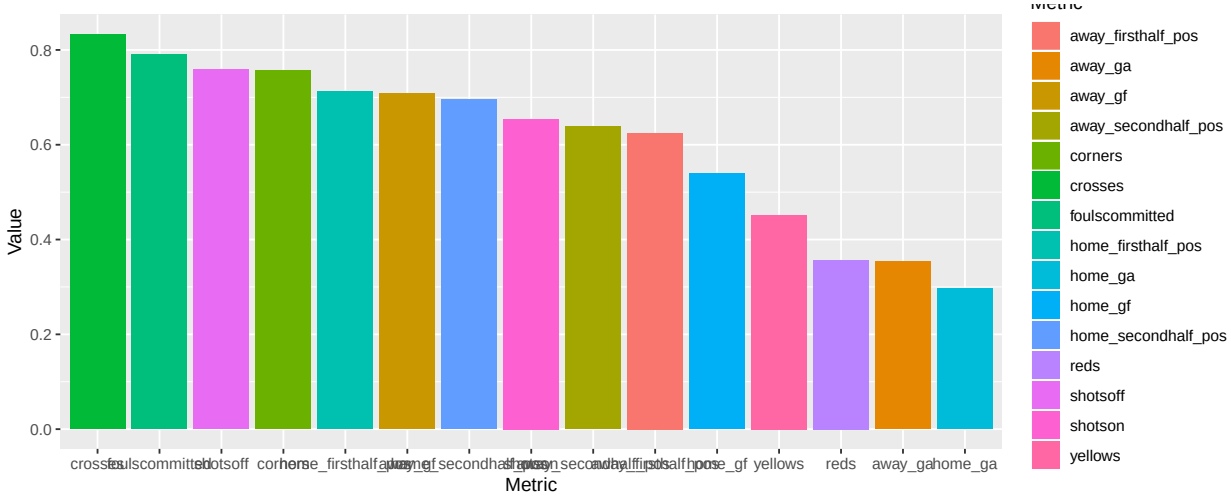
```
## # A tibble: 1 x 15
```

```
##   home_gf home_ga away_gf away_ga shotsoff shotson foulscommitted yellows
##     <dbl>   <dbl>   <dbl>   <dbl>    <dbl>   <dbl>          <dbl>   <dbl>
## 1   0.540   0.297   0.708   0.354    0.758   0.654          0.790   0.452
## # ... with 7 more variables: reds <dbl>, corners <dbl>, crosses <dbl>,
## #   home_firsthalf_pos <dbl>, home_secondhalf_pos <dbl>,
## #   away_firsthalf_pos <dbl>, away_secondhalf_pos <dbl>
```

```
#=================================
#plot clusters
#add cluster for plotting

metrics = c("Home GF", "Home GA", "Away GF","Away GA","Shots Off Target", "Shots On Target",
            "Fouls Committed","Yellow Cards","Red Cards","Corners","Crosses",
            "Home 1st Half Pos.","Home 2nd Half Pos.",
            "Away 1st Half Pos.","Away 2nd Half Pos.")

cluster.stats = as.data.frame(cluster.stats) %>% mutate(cluster = row_number())

cluster.stats %>%
  gather("Metric","Value",-cluster) %>%
  ggplot(aes(x = Metric, y = Value, fill = Metric)) +
  geom_col() +
  facet_grid(rows = vars(cluster))
```




```
#this is what the team's 2015/2016 season looks like.
data_normalized[tail(roma_rows, n=1),] %>%
  gather("Metric","Value") %>%
  ggplot(aes(x = reorder(Metric, -Value), y = Value, fill = Metric)) +
  geom_col() + labs(x = "Metric", y = "Value")
```



2. "Top Four" vs. AS Roma Team Attribute Comparison

This is a base table for different types of analyses. We can use this to get counts of winners by Season/League - who are top performing teams?

```
## Get country and league names and have a final master match table

master_df <- sqldf("SELECT m.id, c.name AS country, l.name AS league, season, date,
home_team, away_team, home_team_goal, away_team_goal, game_result, total_goals,
                   game_winner, game_loser, winner_goals, loser_goals, year, month
                   FROM match_away_teams m
                   INNER JOIN country_df c
                   ON m.country_id = c.id
                   INNER JOIN league_df l
                   ON m.league_id = l.id")

head(master_df) %>% knitr::kable()
```

id	country	league	season	date	home_team	away_team	home_team_
1	Belgium	Belgium Jupiler League	2008/2009	2008-08-17	KRC Genk	Beerschot AC	
2	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	SV Zulte-Waregem	Sporting Lokeren	
3	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	KSV Cercle Brugge	RSC Anderlecht	
4	Belgium	Belgium Jupiler League	2008/2009	2008-08-17	KAA Gent	RAEC Mons	
5	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	FCV Dender EH	Standard de Liège	
6	Belgium	Belgium Jupiler League	2008/2009	2008-09-24	KV Mechelen	Club Brugge KV	

```
## Show the top four leagues - by home winners
best_home_league[1:4,] %>% knitr::kable()
```

league	n
England Premier League	160
France Ligue 1	160
Italy Serie A	160

league	n
Spain LIGA BBVA	160

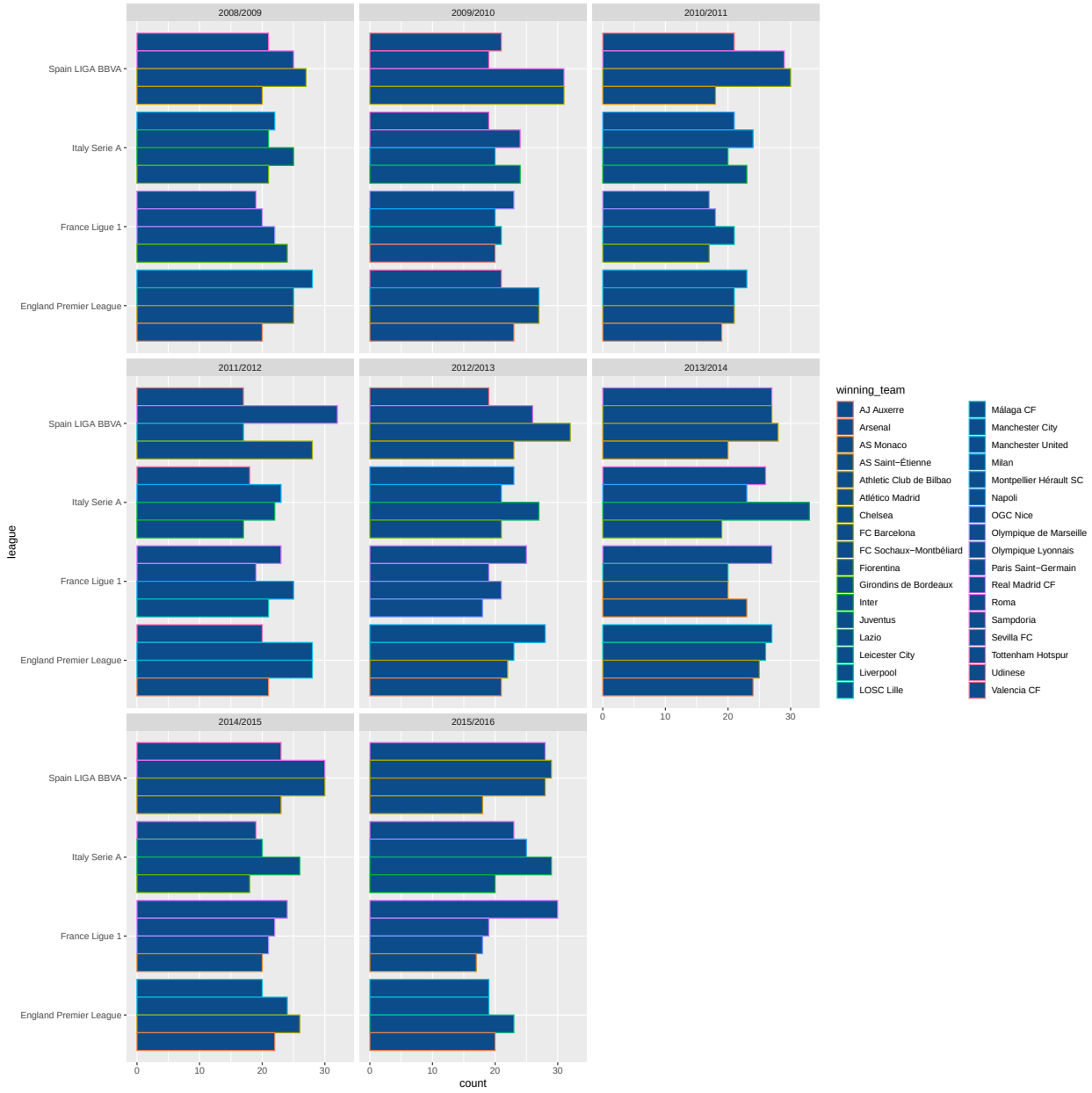
```
## Show the top four leagues - by away winners
best_away_league[1:4,] %>% knitr::kable()
```

league	n
England Premier League	160
France Ligue 1	160
Italy Serie A	160
Spain LIGA BBVA	160

The leagues with the most home and away team wins are England Premier League, France Ligue 1, Italy Serie A, and Spain LIGA BBVA, we will focus our team level analyses solely on these leagues. This leagues currently have the best teams, so if we can target the 4 highest performing teams in each league (16 teams total), we can gain insights into what allows this teams to excel.

```
##      season                 league           result
## 1 2008/2009 England Premier League              tie
## 2 2008/2009 England Premier League          Arsenal
## 3 2008/2009 England Premier League        Liverpool
## 4 2008/2009 England Premier League  West Ham United
## 5 2008/2009 England Premier League      Aston Villa
## 6 2008/2009 England Premier League Blackburn Rovers
```

```
ggplot(top_four_teams) +
  aes(x = league, colour = winning_team, weight = num_wins) +
  geom_bar(position = "dodge", fill = "#0c4c8a") +
  scale_color_hue() +
  coord_flip() +
  theme_gray() +
  facet_wrap(vars(season))
```



OK we have a list of teams that are consistently peforming well, placing the top four in each of our identified leagues. Next we will try to identify want to see what attributes/characteristics comprise the make-up of the team.

```
head(team_att_season_df) %>% knitr::kable()
```

id	team_fifa_api_id	team_api_id	team	buildUpPlaySpeed	buildUpPlaySpeedClass	build
107	57	8583	AJ Auxerre	30	Slow	
836	69	9829	AS Monaco	35	Balanced	
1119	1819	9853	AS Saint-Étienne	65	Balanced	
71	1	9825	Arsenal	66	Balanced	
89	448	8315	Athletic Club de Bilbao	60	Balanced	
95	240	9906	Atlético Madrid	64	Balanced	

Data Preparation

1. Create three distinct categories for team attributes: Attack, Midfield, Defense
2. Subset each category into it's own table for easier analysis
3. View the attributes for each category

Some samples of the tables and the attributes they contain

```
attack_df <- team_att_season_df[, attack]

midfield_df <- team_att_season_df[, midfield]

defense_df <- team_att_season_df[, defense]

head(attack_df) %>% knitr::kable()
```

buildUpPlaySpeed	buildUpPlaySpeedClass	buildUpPlayPassing	buildUpPlayPassingClass	buildUpPlayPositioningCl
30	Slow	70	Long	Organised
35	Balanced	55	Mixed	Organised
65	Balanced	65	Mixed	Organised
66	Balanced	30	Short	Free Form
60	Balanced	45	Mixed	Organised
64	Balanced	30	Short	Free Form

```
head(midfield_df) %>% knitr::kable()
```

chanceCreationPassing	chanceCreationPassingClass	chanceCreationCrossing	chanceCreationCrossingClass	chanceCr
35	Normal	65	Normal	
65	Normal	55	Normal	
55	Normal	60	Normal	
30	Safe	45	Normal	
35	Normal	70	Lots	
65	Normal	50	Normal	

```
head(defense_df) %>% knitr::kable()
```

defencePressure	defencePressureClass	defenceAggression	defenceAggressionClass	defenceTeamWidth	defenceTeamW
30	Deep	55	Press	30	Narrow
70	High	65	Press	70	Wide
30	Deep	30	Contain	30	Narrow
30	Deep	40	Press	50	Normal
30	Deep	70	Double	35	Normal
70	High	34	Press	55	Normal

Summary of Top Four Teams Attacking Attributes

```
## Summary of top four teams' offensive numeric attributes
## Average build up play speed is 52/100 and passing (accuracy) is 45/100 - Roma can do better here!
summary(dplyr::select_if(attack_df, is.numeric)) %>% knitr::kable()
```

buildUpPlaySpeed	buildUpPlayPassing
Min. :20.00	Min. :20.0
1st Qu.:45.00	1st Qu.:36.0
Median :50.50	Median :47.0
Mean :52.46	Mean :45.6
3rd Qu.:63.25	3rd Qu.:52.0
Max. :78.00	Max. :70.0

```
## Balanced speed for building up play, mix of long/short passing for build up play
## build up position is organized - teams that are successful play with balance (can we exploit this wi
dplyr::select_if(attack_df, is.character) %>%
  group_by(buildUpPlaySpeedClass, buildUpPlayPassingClass, buildUpPlayPositioningClass) %>%
  summarise(Freq = n()) %>%
  arrange(-Freq) %>% knitr::kable()
```

buildUpPlaySpeedClass	buildUpPlayPassingClass	buildUpPlayPositioningClass	Freq
Balanced	Mixed	Organised	132
Fast	Mixed	Organised	17
Balanced	Mixed	Free Form	13
Balanced	Short	Organised	10
Balanced	Short	Free Form	8
Fast	Long	Organised	5
Fast	Mixed	Free Form	4
Fast	Short	Free Form	3
Slow	Mixed	Organised	3
Slow	Short	Free Form	3
Slow	Short	Organised	3
Balanced	Long	Organised	1
Fast	Short	Organised	1
Slow	Long	Organised	1

Average (mean) build up play speed is 52.46 for top performing teams and build up passing is 45.6. This points toward top performing teams having a "balanced" build up play style.

Summary of Toup Four Teams Midfield Attributes

```
## Summary of top four teams' midfield numeric attributes
summary(dplyr::select_if(midfield_df, is.numeric)) %>% knitr::kable()
```

chanceCreationPassing	chanceCreationCrossing	chanceCreationShooting
Min. :28.00	Min. :20.00	Min. :23.00
1st Qu.:46.00	1st Qu.:50.00	1st Qu.:50.00
Median :50.50	Median :56.00	Median :54.50
Mean :53.19	Mean :55.58	Mean :55.75

chanceCreationPassing	chanceCreationCrossing	chanceCreationShooting
3rd Qu.:65.00	3rd Qu.:65.00	3rd Qu.:67.00
Max. :77.00	Max. :80.00	Max. :80.00

```
## Successful teams create chances from midfield normally, from cross normal, and shooting opportunitie
## And are organized in there positioning.  Nothing too suprising about that.
dplyr::select_if(midfield_df, is.character) %>%
  group_by(chanceCreationPassingClass, chanceCreationCrossingClass, chanceCreationShootingClass, chance
  summarise(Freq = n()) %>%
  arrange(-Freq) %>% knitr::kable()
```

chanceCreationPassingClass	chanceCreationCrossingClass	chanceCreationShootingClass	chanceCreationPositioningCla
Normal	Normal	Normal	Organised
Normal	Normal	Lots	Organised
Normal	Normal	Normal	Free Form
Normal	Lots	Normal	Organised
Normal	Normal	Lots	Free Form
Risky	Lots	Normal	Organised
Risky	Normal	Lots	Free Form
Risky	Normal	Normal	Organised
Risky	Normal	Lots	Organised
Normal	Little	Normal	Free Form
Risky	Normal	Normal	Free Form
Normal	Lots	Lots	Organised
Normal	Normal	Little	Organised
Safe	Normal	Normal	Free Form
Normal	Lots	Normal	Free Form
Normal	Normal	Little	Free Form
Risky	Lots	Lots	Free Form
Risky	Lots	Normal	Free Form
Normal	Little	Lots	Free Form
Normal	Little	Normal	Organised
Normal	Lots	Little	Organised
Normal	Lots	Lots	Free Form
Risky	Lots	Lots	Organised
Safe	Little	Normal	Free Form
Safe	Normal	Lots	Free Form
Safe	Normal	Lots	Organised
Safe	Normal	Normal	Organised

When looking at the midfield categories, we see that passing on average is 53.19, crossing is 55.58, and shooting is 55.75. This data indicates that successful teams are utilizing their midfield effectively to create chances and are organized in their positioning. This is not too surprising because we expect top teams to be utilizing the midfield when playing.

Summary of Toup Four Teams Defense Attributes

```
## Summary of top four teams' defense numeric attributes
summary(dplyr::select_if(defense_df, is.numeric)) %>% knitr::kable()
```

defencePressure	defenceAggression	defenceTeamWidth
Min. :23.00	Min. :30.00	Min. :30.00
1st Qu.:40.00	1st Qu.:45.00	1st Qu.:49.00
Median :49.00	Median :49.00	Median :53.50
Mean :48.36	Mean :50.67	Mean :53.39
3rd Qu.:56.00	3rd Qu.:57.00	3rd Qu.:60.00
Max. :70.00	Max. :70.00	Max. :70.00

```
dplyr::select_if(defense_df, is.character) %>%
  group_by(defencePressureClass, defenceAggressionClass, defenceTeamWidthClass, defenceDefenderLineClass
  summarise(Freq = n()) %>%
  arrange(-Freq) %>% knitr::kable()
```

defencePressureClass	defenceAggressionClass	defenceTeamWidthClass	defenceDefenderLineClass	Freq
Medium	Press	Normal	Cover	151
Medium	Press	Normal	Offside Trap	12
Medium	Press	Wide	Cover	6
Deep	Press	Normal	Cover	5
Medium	Double	Normal	Cover	5
Medium	Contain	Normal	Offside Trap	3
Deep	Double	Normal	Cover	2
High	Double	Wide	Cover	2
High	Double	Wide	Offside Trap	2
High	Press	Wide	Cover	2
Medium	Contain	Normal	Cover	2
Medium	Double	Normal	Offside Trap	2
Deep	Contain	Narrow	Offside Trap	1
Deep	Press	Narrow	Cover	1
Deep	Press	Normal	Offside Trap	1
High	Contain	Wide	Offside Trap	1
High	Press	Normal	Cover	1
High	Press	Normal	Offside Trap	1
High	Press	Wide	Offside Trap	1
Medium	Contain	Wide	Offside Trap	1
Medium	Double	Wide	Offside Trap	1
Medium	Press	Wide	Offside Trap	1

Top performing teams have an average defense pressure of 48.36, aggression of 50.67, and team width of 53.39. Again, nothing too surprising here about how top defensive groups are playing.

Now that we've analyzed successful teams, we want to see if there is area for improvement, either in attack, midfield, or defense for AS Roma. To do that, we need to look at their team attributes.

Summary of AS Roma Attacking Attributes

```
## Summary of offensive numeric attributes
summary(dplyr::select_if(roma_attack_df, is.numeric)) %>% knitr::kable()
```

buildUpPlaySpeed	buildUpPlayPassing
Min. :53.00	Min. :30.00
1st Qu.:61.00	1st Qu.:32.00
Median :65.50	Median :39.00
Mean :63.67	Mean :40.00
3rd Qu.:67.75	3rd Qu.:45.25
Max. :70.00	Max. :55.00

```
## Roma tends to play fast, with a mixed set of passing, but keep an organized shape
dplyr::select_if(roma_attack_df, is.character) %>%
  group_by(buildUpPlaySpeedClass, buildUpPlayPassingClass, buildUpPlayPositioningClass) %>%
  summarise(Freq = n()) %>%
  arrange(-Freq) %>% knitr::kable()
```

buildUpPlaySpeedClass	buildUpPlayPassingClass	buildUpPlayPositioningClass	Freq
Fast	Mixed	Organised	2
Balanced	Mixed	Free Form	1
Balanced	Mixed	Organised	1
Balanced	Short	Organised	1
Fast	Short	Free Form	1

AS Roma has an average play speed of 63.75 and a passing accuracy of 40.00. Top performing teams have measures of 52.46 and 45.6 respectively. This is indicating that compared to top teams, AS Roma is playing quicker and missing more of their passes. AS Roma should consider slowing down their play speed and ensure that passes are being made accurately and deliberately to exploit opportunities on the field.

Summary of AS Roma Midfield Attributes

```
summary(dplyr::select_if(roma_midfield_df, is.numeric))  %>% knitr::kable()
```

chanceCreationPassing	chanceCreationCrossing	chanceCreationShooting
Min. :60.00	Min. :35.00	Min. :50.00
1st Qu.:65.50	1st Qu.:51.00	1st Qu.:56.00
Median :69.00	Median :57.00	Median :60.50
Mean :68.83	Mean :56.33	Mean :60.50
3rd Qu.:72.50	3rd Qu.:63.00	3rd Qu.:65.75
Max. :77.00	Max. :75.00	Max. :70.00

```
## Risky chance creation, free form creation / positioning - fluid midfield
dplyr::select_if(roma_midfield_df, is.character) %>%
  group_by(chanceCreationPassingClass, chanceCreationCrossingClass,
           chanceCreationShootingClass, chanceCreationPositioningClass) %>%
      summarise(Freq = n()) %>%
          arrange(-Freq)  %>% knitr::kable()
```

chanceCreationPassingClass	chanceCreationCrossingClass	chanceCreationShootingClass	chanceCreationPositioningCla
Risky	Normal	Normal	Free Form
Normal	Normal	Lots	Free Form
Normal	Normal	Normal	Free Form
Risky	Lots	Normal	Free Form

AS Roma has an average passing creation of 68.83, crossing creation of 56.33, and shooting creation of 60.50. Top performing teams have metrics of 53.19, 55.58, and 55.75 respectively. This indicates that AS Roma is actually playing slightly better on average in the midfield than top performing teams.

Summary of AS Roma Defense Attributes

```
## Summary of defensive numeric attributes
## Low pressure, low agression, 50/100 team width
summary(dplyr::select_if(roma_defense_df, is.numeric)) %>% knitr::kable()
```

defencePressure	defenceAggression	defenceTeamWidth
Min. :35.00	Min. :30.00	Min. :34.00
1st Qu.:36.00	1st Qu.:45.50	1st Qu.:48.50
Median :41.00	Median :47.00	Median :50.00
Mean :42.83	Mean :47.17	Mean :50.83
3rd Qu.:48.25	3rd Qu.:54.50	3rd Qu.:52.25
Max. :55.00	Max. :57.00	Max. :70.00

```
## Summary of defensive categorical attributes
## they play the press, normal team shape and a high line due to their offside trap scheme
# - maybe this is an area for improvement?
dplyr::select_if(roma_defense_df, is.character) %>%
  group_by(defencePressureClass, defenceAggressionClass, defenceTeamWidthClass, defenceDefenderLineClass
  summarise(Freq = n()) %>%
  arrange(-Freq) %>% knitr::kable()
```

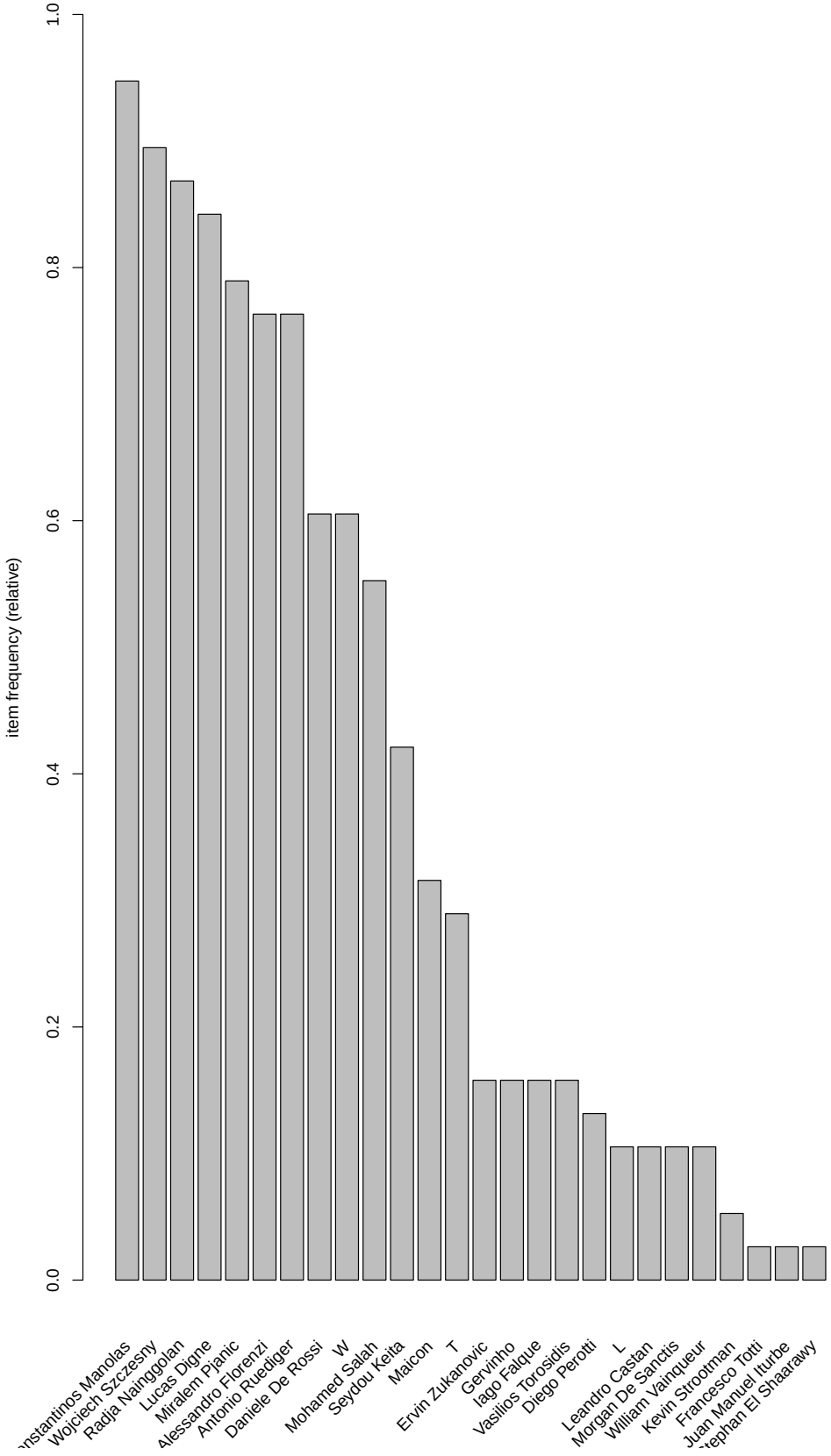
defencePressureClass	defenceAggressionClass	defenceTeamWidthClass	defenceDefenderLineClass	Freq
Medium	Press	Normal	Offside Trap	3
Medium	Press	Normal	Cover	2
Medium	Contain	Wide	Offside Trap	1

AS Roma has an average defense pressure of 42.83, average aggression of 47.17, and average team width of 50.83. Top teams have metrics of 48.36, 50.67, and 53.39 respectively. This indicates AS Roma should increase its pressure and aggression on defense to mimic how top performing teams often play defense.

Now that we've seen how AS Roma compares to our "competitors" in the top four of the Premier League, Serie A, La Liga, and Ligue Une, let's see if we can dig into player attributes for Roma and what makes them stand out?

3. Player Analysis based on 2015/2016 season for what players should be included in team of future

```
# Grab the all_data CSV using your working directory (have it all in one place)
csv_file <- "all_data.csv"

all_data = read.transactions(csv_file, format = "basket",
                             sep = ",", rm.duplicates = TRUE, header = TRUE)

itemFrequencyPlot(all_data, ylim = c(0, 1), topN = 30)
```

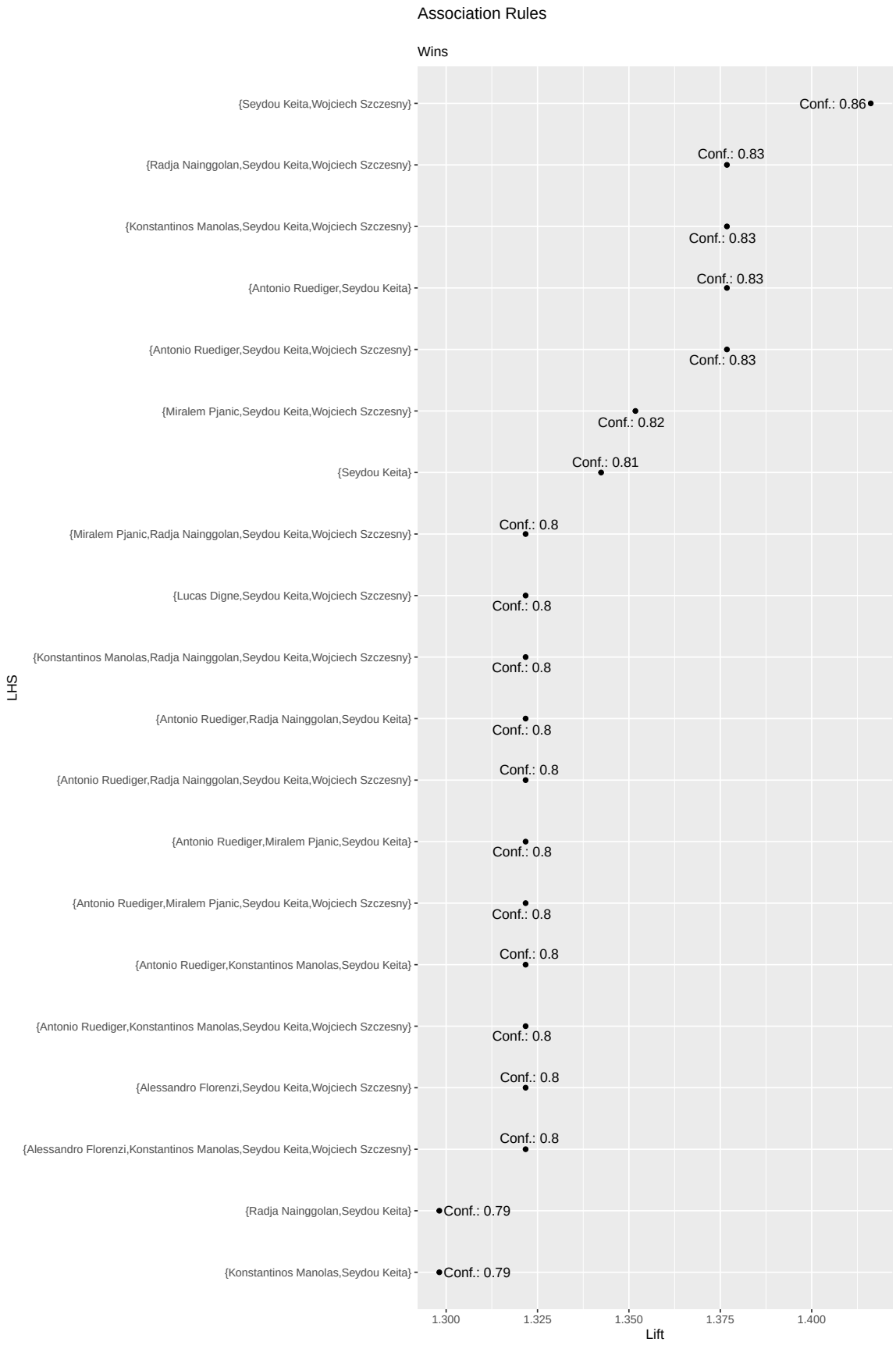

item frequency (relative)
0.0
0.2
0.4
0.6
0.8
1.0
nstantinos Manolas
Wojciech Szczesny
Radja Nainggolan
Lucas Digne
Miralem Pjanic
Alessandro Florenzi
Antonio Ruediger
Daniele De Rossi
W
Mohamed Salah
Seydou Keita
Maicon
T
Ervin Zukanovic
Gervinho
Iago Falque
Vasilios Torosidis
Diego Perotti
L
Leandro Castan
Morgan De Sanctis
William Vainqueur
Kevin Strootman
Francesco Totti
Juan Manuel Iturbe
tephan El Shaarawy

```r
results <- data.frame()
for (n in seq(from=0.7, to=1.0, by=0.1)) {
  rules <- apriori(all_data, parameter=list(supp=0.2, conf=n, target="rules", minlen=2))
  dfxx = data.frame(lhs=labels(lhs(rules)), rhs=labels(rhs(rules)),rules@quality)
  results <- rbind(results, dfxx)
}
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    0.1    1 none FALSE            TRUE       5     0.2      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [13 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 done [0.00s].
## writing ... [2248 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.2      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [13 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 done [0.00s].
## writing ... [1878 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
```

```
##         0.9    0.1    1 none FALSE            TRUE       5     0.2      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [13 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 done [0.00s].
## writing ... [972 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##           1    0.1    1 none FALSE            TRUE       5     0.2      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 7
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [13 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 done [0.00s].
## writing ... [497 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
xresults <- results %>% filter(rhs == "{W}")
xresults <- xresults[(!duplicated(xresults)), ]
xresults2 <- head(xresults %>% arrange(desc(xresults$lift)), 20)
ggplot(xresults2, aes(x=reorder(xresults2$lhs, xresults2$lift), y=xresults2$lift)) +
  geom_point() + coord_flip() +
  labs(title="Association Rules\n", subtitle="Wins",x="LHS", y="Lift") +
  ggrepel::geom_text_repel(label=paste0("Conf.: ", round(xresults2$confidence,2)))
```


Association Rules
Wins
{Seydou Keita,Wojciech Szczesny}
{Radja Nainggolan,Seydou Keita,Wojciech Szczesny}
{Konstantinos Manolas,Seydou Keita,Wojciech Szczesny}
{Antonio Ruediger,Seydou Keita}
{Antonio Ruediger,Seydou Keita,Wojciech Szczesny}
{Miralem Pjanic,Seydou Keita,Wojciech Szczesny}
{Seydou Keita}
{Miralem Pjanic,Radja Nainggolan,Seydou Keita,Wojciech Szczesny}
{Lucas Digne,Seydou Keita,Wojciech Szczesny}
{Konstantinos Manolas,Radja Nainggolan,Seydou Keita,Wojciech Szczesny}
{Antonio Ruediger,Radja Nainggolan,Seydou Keita}
{Antonio Ruediger,Radja Nainggolan,Seydou Keita,Wojciech Szczesny}
{Antonio Ruediger,Miralem Pjanic,Seydou Keita}
{Antonio Ruediger,Miralem Pjanic,Seydou Keita,Wojciech Szczesny}
{Antonio Ruediger,Konstantinos Manolas,Seydou Keita}
{Antonio Ruediger,Konstantinos Manolas,Seydou Keita,Wojciech Szczesny}
{Alessandro Florenzi,Seydou Keita,Wojciech Szczesny}
{Alessandro Florenzi,Konstantinos Manolas,Seydou Keita,Wojciech Szczesny}
{Radja Nainggolan,Seydou Keita}
{Konstantinos Manolas,Seydou Keita}
LHS
Conf.: 0.86
Conf.: 0.83
Conf.: 0.83
Conf.: 0.83
Conf.: 0.83
Conf.: 0.82
Conf.: 0.81
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.8
Conf.: 0.79
Conf.: 0.79
1.300
1.325
1.350
1.375
1.400
Lift

4. Recommendations

Final Takeaways:

- Winning teams increase crosses, shots taken, corners, and ball possession time
- Winning teams decrease penalties
- Winning teams have offensive play styles that are slightly slower than AS Roma, but more precise with their passing
- Winning teams have defensive play styles that are slightly more aggressive and add more pressure than AS Roma
- AS Roma has a strong midfield and should capitalize on that as a key element of their play style

Final Recommendation:

AS Roma has a few opportunities to greatly improve performance and be more successful within its league. First, from a team strategy, AS Roma should focus more on precise and accurate passing while slowing down play on oense. This will allow the game to progress and expose vulnerabilities in the opposing team that can be exploited. AS Roma should keep its midfield play style similar to how it already is. Lastly, AS Roma should increase aggression and pressure on defense. This will create opportunities for take-aways and allow less shots on goal. In specific aspects of AS Roma's play, the team should focus on increasing the number of crosses, shots taken, possession time, and corner kicks. Though obvious, best teams utilize these aspects of their game to take advantage of other teams. Also, AS Roma should reduce penalties. Successful teams oen have low numbers of penalties in their games. Lastly, we used our association rules analysis to determine specific players that played together that led to increases in AS Roma's likelihood of winning during the most recent season of currently available data. Players like Konstantinos Manolas (defender), Radja Nainnggolan (defender), Seydou Keita (midfielder), and Wojciech Szczesny (goalkeeper) likely exhibit attributes of players who will lead AS Roma to success in future seasons.