Homework 2: Association Rules and Sports Analytics

Tarun Newton, Abhinav Kumar, Pranav Lingaraju, Kavya Puthuvaya, Pranav Srinivas Nagavelli, Sarath Haridas

3 October 2018

Contents

A. The Business Problem

Chelsea FC has finished 10th in the English Premier League in the 2015/2016 season from being a winner in the previous season. As, the resident Data Scientists of the team, we have used analytics to identify patterns that were prevalent in Chelsea's success.

The complication was to identify non-obvious patterns that impacted our success.

Football like every industry is becoming increasingly data-driven and there exists an enormous advantage to be leveraged by generating the right kind of insights from the data collected at Chelsea. Football, as we know, is very competitive so even a small improvement can result in a huge difference in results. Chelsea is valued at $2 billion and brings in a revenue of $466M yearly and this hinges on the success of our players every season.

1. Why Association Rules?

We believe that association rules can be leveraged because Football is a weak link sport. Sally, a behavioral economist through statistical analysis shows in his book, The Numbers Game, that teams win more games if they improve the weakest player rather than improving their strongest player.

We first plan to explore the data we have at hand to first understand the different kinds of analysis and then dive into association rules to see if we can find some non-obvious patterns that we can leverage to help Chelsea achieve success the coming year!

References:
https://www.forbes.com/teams/chelsea/
http://revisionisthistory.com/episodes/06-my-little-hundred-million (Malcolm Gladwell's Podcast)
https://www.theguardian.com/books/2013/may/24/numbers-game-everything-football-wrong

B. Exploratory Data Analysis

Load Dependencies

```
library(dplyr)
library(arules)
library(tidyr)
library(RSQLite)
library(knitr)
library(ggplot2)
library(scales)
library(ggthemes)
library(ggrepel)
library(magrittr)
library(VIM)
library(corrplot)
library(stats)
library(data.table)
library(datasets)
library(arulesViz)
library(gridExtra)
library(reshape2)
library(pander)
library(viridis)
library(xml2)
library(purrr)
library(tibble)
```

1. Load Data

Loading the data hosted in the SQL database

```
con <- src_sqlite("euro_soccer.sqlite")

country_tbl <- tbl(con, "country")
country = collect(country_tbl)

league_tbl <- tbl(con, "league")
league = collect(league_tbl)

match_tbl <- tbl(con, "match")
match = collect(match_tbl)

player_tbl <- tbl(con, "player")
player = collect ( player_tbl)

player_atts_tbl <- tbl(con, "player_attributes")
player_atts = collect (player_atts_tbl)

team_tbl <- tbl(con, "team")
team = collect(team_tbl)
```

2. Data Exploration

We see that we have 6 tables in the SQL database which we can leverage for our analysis. We'll now look at each of the tables to understand what information is present and if any data issues exist.

Country Table

```
country
```

```
## # A tibble: 11 x 2
##       id name
##    <int> <chr>
##  1     1 Belgium
##  2  1729 England
##  3  4769 France
##  4  7809 Germany
##  5 10257 Italy
##  6 13274 Netherlands
##  7 15722 Poland
##  8 17642 Portugal
##  9 19694 Scotland
## 10 21518 Spain
## 11 24558 Switzerland
```

In the country table we see that there is data from 11 different countries. The league in England is what we are interested in as Chelsea plays in the English Premier League. We can use that information to filter the other tables for the league in England.

League Table

```
league
```

```
## # A tibble: 11 x 3
##       id country_id name
##    <int>      <int> <chr>
##  1     1          1 Belgium Jupiler League
##  2  1729       1729 England Premier League
##  3  4769       4769 France Ligue 1
##  4  7809       7809 Germany 1. Bundesliga
##  5 10257      10257 Italy Serie A
##  6 13274      13274 Netherlands Eredivisie
##  7 15722      15722 Poland Ekstraklasa
##  8 17642      17642 Portugal Liga ZON Sagres
##  9 19694      19694 Scotland Premier League
## 10 21518      21518 Spain LIGA BBVA
## 11 24558      24558 Switzerland Super League
```

In the league table we see that we have the league names for the 11 countries that are present. As mentioned, Chelsea plays in the English Premier League (League ID: 1729)

Match Table

```
dim(match)
```

```
## [1] 25979   115
```

From our exploration we see that the match table contains all the information regarding a certain match from seasons 2008 to 2016. Due to its large size we have displayed the summary of this table in the Appendix. It also has betting odds from up to 10 providers and detailed match events (goal types, possession, corner, cross, fouls, cards etc…)

Also, at a high level we see that there are nulls in the overall table. When we filter for Chelsea we see that there are no nulls or any outlier information in the player information for each match so we feel confident in using the table as is.

Player Table

```
summary(player)
```

```
##        id         player_api_id    player_name        player_fifa_api_id
##  Min.   :    1   Min.   :  2625   Length:11060       Min.   :     2
##  1st Qu.: 2768   1st Qu.: 35556   Class :character   1st Qu.:151890
##  Median : 5536   Median : 96620   Mode  :character   Median :184671
##  Mean   : 5538   Mean   :156582                      Mean   :165665
##  3rd Qu.: 8306   3rd Qu.:212471                      3rd Qu.:203883
##  Max.   :11075   Max.   :750584                      Max.   :234141
##    birthday             height          weight
##  Length:11060       Min.   :157.5   Min.   :117.0
##  Class :character   1st Qu.:177.8   1st Qu.:159.0
##  Mode  :character   Median :182.9   Median :168.0
##                     Mean   :181.9   Mean   :168.4
##                     3rd Qu.:185.4   3rd Qu.:179.0
##                     Max.   :208.3   Max.   :243.0
```

```
glimpse(player)
```

```
## Observations: 11,060
## Variables: 7
## $ id                 <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ...
## $ player_api_id      <int> 505942, 155782, 162549, 30572, 23780, 27316...
## $ player_name        <chr> "Aaron Appindangoye", "Aaron Cresswell", "A...
## $ player_fifa_api_id <int> 218353, 189615, 186170, 140161, 17725, 1581...
## $ birthday           <chr> "1992-02-29 00:00:00", "1989-12-15 00:00:00...
## $ height             <dbl> 182.88, 170.18, 170.18, 182.88, 182.88, 182...
## $ weight             <int> 187, 146, 163, 198, 154, 161, 146, 139, 181...
```

In the Player table we have all the player attributes which are sourced from EA Sports' FIFA video game series.

Player Atts Table

```
dim(player_atts)
```

```
## [1] 183978     42
```

On our exploration of the player atts table we see that it has different player attributes which are sourced from EA Sports' FIFA video game series. These metrics are interesting and will help us quantify the player belyond his name and club. Due to its size we have displayed the summary in the appendix.

Team Table

```
summary(team)
```

```
##        id         team_api_id     team_fifa_api_id   team_long_name
##  Min.   :    1   Min.   :  1601   Min.   :     1.0   Length:299
##  1st Qu.: 9552   1st Qu.:  8349   1st Qu.:   178.8   Class :character
##  Median :22805   Median :  8655   Median :   673.5   Mode  :character
##  Mean   :23735   Mean   : 12341   Mean   : 21534.3
##  3rd Qu.:36251   3rd Qu.:  9886   3rd Qu.:  1910.8
##  Max.   :51606   Max.   :274581   Max.   :112513.0
```

```
##                                 NA's   :11
##   team_short_name
##   Length:299
##   Class :character
##   Mode  :character
##
##
##
##
```

```r
glimpse(team)
```

```
## Observations: 299
## Variables: 5
## $ id               <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14...
## $ team_api_id      <int> 9987, 9993, 10000, 9994, 9984, 8635, 9991, 99...
## $ team_fifa_api_id <int> 673, 675, 15005, 2007, 1750, 229, 674, 1747, ...
## $ team_long_name   <chr> "KRC Genk", "Beerschot AC", "SV Zulte-Waregem...
## $ team_short_name  <chr> "GEN", "BAC", "ZUL", "LOK", "CEB", "AND", "GE...
```

In the team table we see it has the team id and name which will help us filter the league table specifically for Chelsea.

3. Overview of Chelsea's League Performance

The Premier League is the top level of the English football league system. Contested by 20 clubs, it operates on a system of promotion and relegation within the English Football League (EFL). Seasons run from August to May with each team playing 38 matches (playing each other home and away).

The final ranking in a season is based on cumulative points where each team get 3 points for a win and 1 for a draw. Below we have calculated the final rankings per season as we wanted to understand how Chelsea has been performing over the last couple of years in the league.

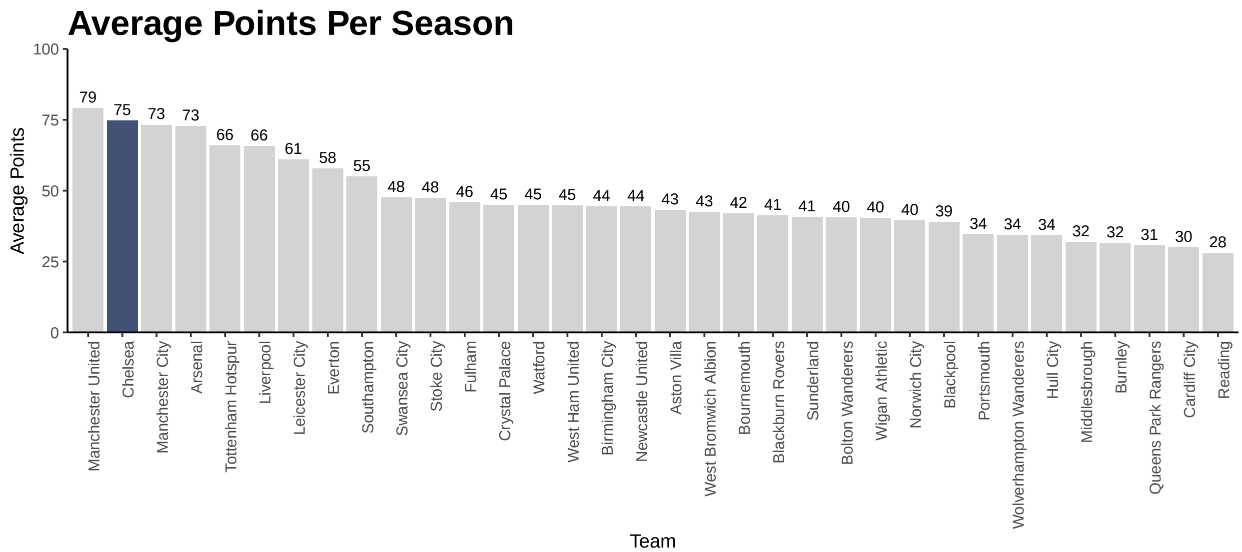
```r
#Creation of the final standings table for each of the 8 seasons

match_points = match %>%  select(1:11)  %>%  mutate(home_point = if_else((home_team_goal >
away_team_goal),3,if_else((home_team_goal == away_team_goal),1,0))) %>% mutate(away_point
= if_else((home_team_goal > away_team_goal),0,if_else((home_team_goal == away_team_goal),
1,3)))
league_table_home = match_points %>%   group_by(season, league_id, home_team_api_id) %>%
summarise(home_points = sum(home_point), home_goals_scored = sum(home_team_goal) ,
home_goals_conceded = sum(away_team_goal))
league_table_away = match_points %>% group_by(season, league_id, away_team_api_id) %>%
summarise(away_points = sum(away_point), away_goals_scored = sum( away_team_goal),
away_goals_conceded = sum(home_team_goal))
league_table = merge ( league_table_home,league_table_away,by.x = c('season', 'league_id',
'home_team_api_id') , by.y = c('season', 'league_id' , 'away_team_api_id') , all.x = TRUE)
league_table = league_table %>% mutate(total_points = home_points + away_points,
total_goals_scored =home_goals_scored+away_goals_scored , total_goals_conceded =
home_goals_conceded + away_goals_conceded ) %>% filter (league_id == 1729 )

# Filter for English Premier League
league_table = merge( league_table , league , by.x = 'league_id' , by.y = 'country_id',
all.x =TRUE  )
league_table = merge( league_table , team , by.x = 'home_team_api_id' ,
by.y = 'team_api_id', all.x =TRUE  )
```

```r
league_table_final = league_table %>% mutate ( team_api_id = home_team_api_id ) %>%
select( c( league_id, name , season,  team_api_id, team_short_name, team_long_name,
total_goals_scored, total_goals_conceded , home_points, away_points , total_points ) ) %>%
group_by(league_id,name,season )  %>% mutate(rank = min_rank(desc(total_points))) %>%
arrange ( league_id,name ,season, rank )
league_table_final = league_table %>% mutate ( team_api_id = home_team_api_id ) %>%
select (c( league_id, name , season,  team_api_id, team_short_name, team_long_name,
total_goals_scored, total_goals_conceded , home_points, away_points,
total_points ) ) %>%  group_by(league_id,name,season )  %>%
mutate(rank = min_rank(desc(total_points))) %>% arrange ( league_id,name ,season,
rank)
league_table_final$chelsea_flag = ifelse(league_table_final$team_short_name=='CHE',1,0)
league_table_final
```

```
## # A tibble: 160 x 13
## # Groups:   league_id, name, season [8]
##    league_id name       season  team_api_id team_short_name team_long_name
##        <int> <chr>      <chr>         <int> <chr>           <chr>
##  1      1729 England P~ 2008/2~       10260 MUN             Manchester Un~
##  2      1729 England P~ 2008/2~        8650 LIV             Liverpool
##  3      1729 England P~ 2008/2~        8455 CHE             Chelsea
##  4      1729 England P~ 2008/2~        9825 ARS             Arsenal
##  5      1729 England P~ 2008/2~        8668 EVE             Everton
##  6      1729 England P~ 2008/2~       10252 AVL             Aston Villa
##  7      1729 England P~ 2008/2~        9879 FUL             Fulham
##  8      1729 England P~ 2008/2~        8586 TOT             Tottenham Hot~
##  9      1729 England P~ 2008/2~        8654 WHU             West Ham Unit~
## 10      1729 England P~ 2008/2~        8456 MCI             Manchester Ci~
## # ... with 150 more rows, and 7 more variables: total_goals_scored <int>,
## #   total_goals_conceded <int>, home_points <dbl>, away_points <dbl>,
## #   total_points <dbl>, rank <int>, chelsea_flag <dbl>
```

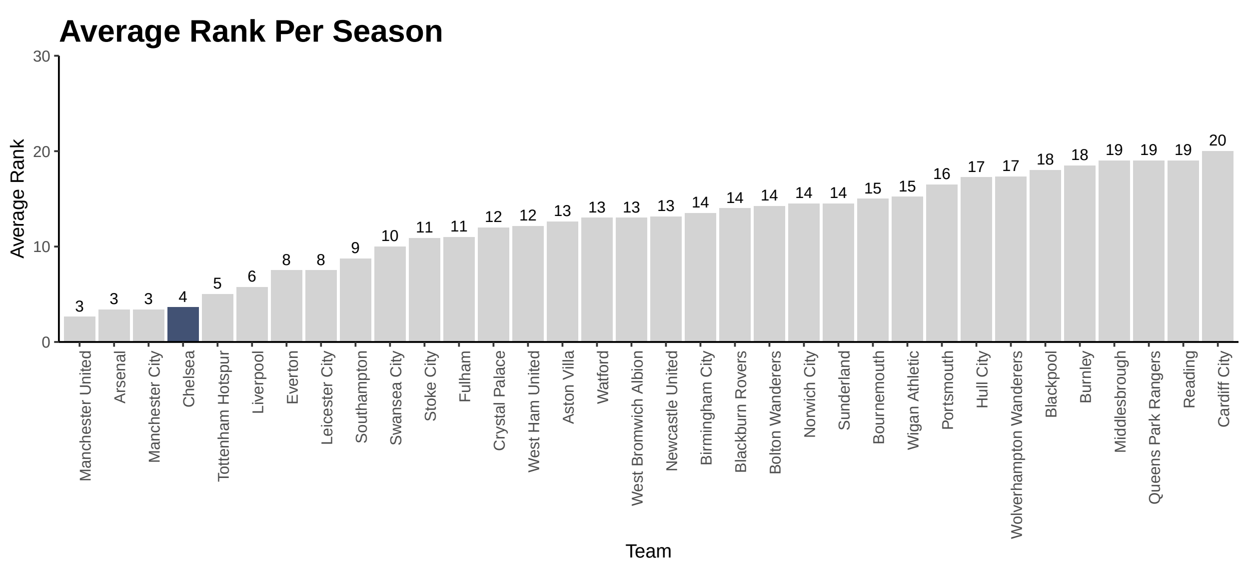
Now that we have calculated the final league table for the 8 seasons, we can analyze Chelsea's performance with respect to the other teams in the Premier League.

```r
#Chelsea league position points , final standing
ggplot (league_table_final, aes ( x = reorder( team_long_name, -total_points) ,
y = total_points, fill =factor (chelsea_flag)) )+stat_summary( fun.y = 'mean' ,
geom  = 'bar')+Theme+theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
scale_fill_manual(guide=F,values=c(col3, col1))+stat_summary(aes(label=round(..y..,0)),
fun.y=mean, geom="text", size=3, vjust = -0.5)+ylim ( 0, 100) +   xlab("Team") +  ylab(
"Average Points") + ggtitle( "Average Points Per Season") +
  theme(plot.title=element_text(size=20,hjust=0,face="bold",colour="black",vjust=1))+
   scale_y_continuous(expand = c(0, 0),limits = c(0, 100))
```



We see that on average Chelsea has 75 points per season which is the second best in the league over the 8 season period.

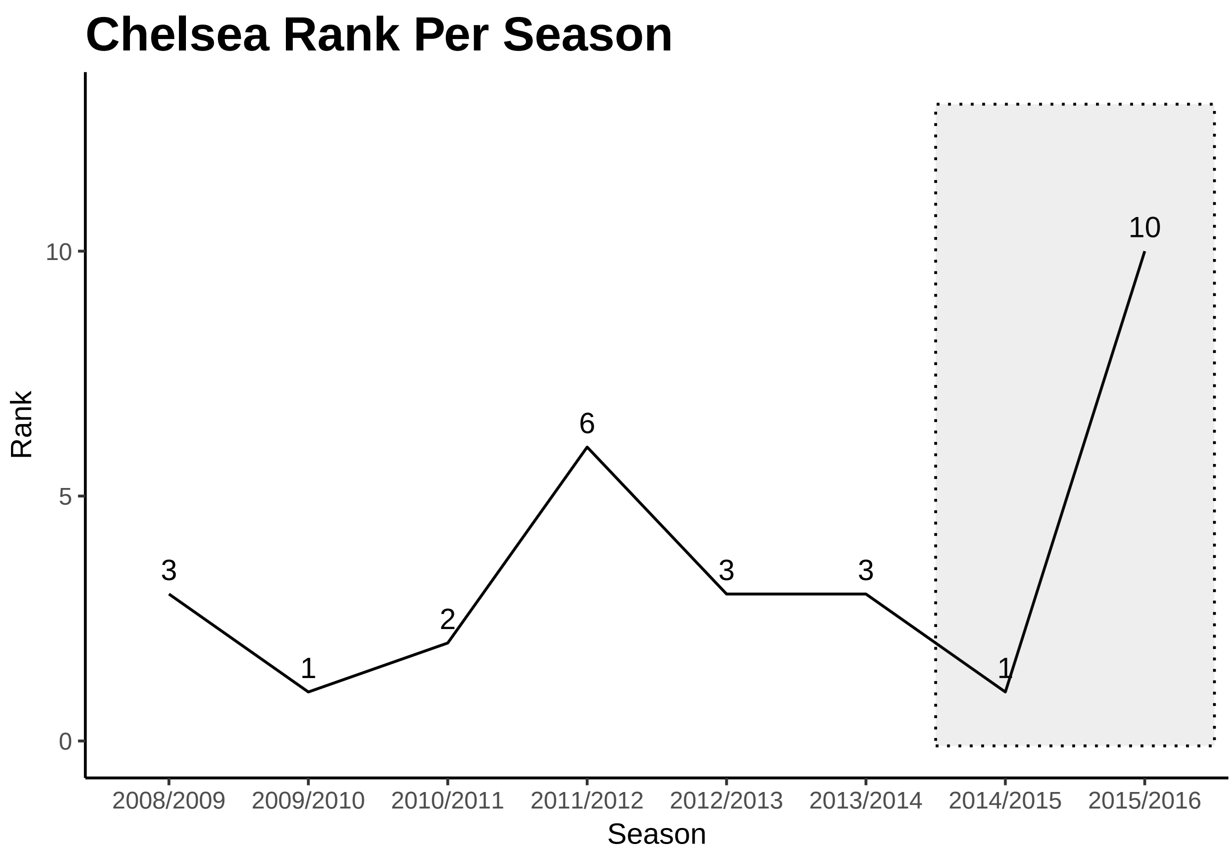
```
ggplot (league_table_final, aes ( x = reorder( team_long_name, rank) , y = rank,
fill =factor (chelsea_flag)) )+ stat_summary( fun.y = 'mean' , geom  = 'bar')+Theme+
theme(axis.text.x = element_text(angle = 90, hjust = 1)) +   scale_fill_manual(
guide=F,values=c(col3, col1))+stat_summary(aes(label=round(..y..,0)),   fun.y=mean,
geom="text", size=3, vjust = -0.5) + ylim ( 0, 30)+xlab("Team")+ylab("Average Rank")+
ggtitle( "Average Rank Per Season")+
  theme(plot.title=element_text(size=18,hjust=0,face="bold",colour="black",vjust=1))+
   scale_y_continuous(expand = c(0, 0),limits = c(0, 30))
```



We see that across the 8 seasons of data that we have, Chelsea's average league position has been 4th. This highlights the fact that Chelsea is one of the top teams in the league.

```
ggplot(subset(league_table_final,team_short_name=='CHE'),aes(x=season,y=rank,label=rank))+
geom_line(group =1)+xlab("Season")+ ylab("Rank")+ggtitle( "Chelsea Rank Per Season")+
scale_y_continuous(breaks= pretty_breaks())+geom_text ( nudge_y = .5 ) +Theme+
annotate("rect",linetype="dotted", xmin=6.5, xmax=8.5, ymin=-.1, ymax=13, color="black",
```

```
  alpha=.1)+
  theme(plot.title=element_text(size=18,hjust=0,face="bold",colour="black",vjust=1))
```

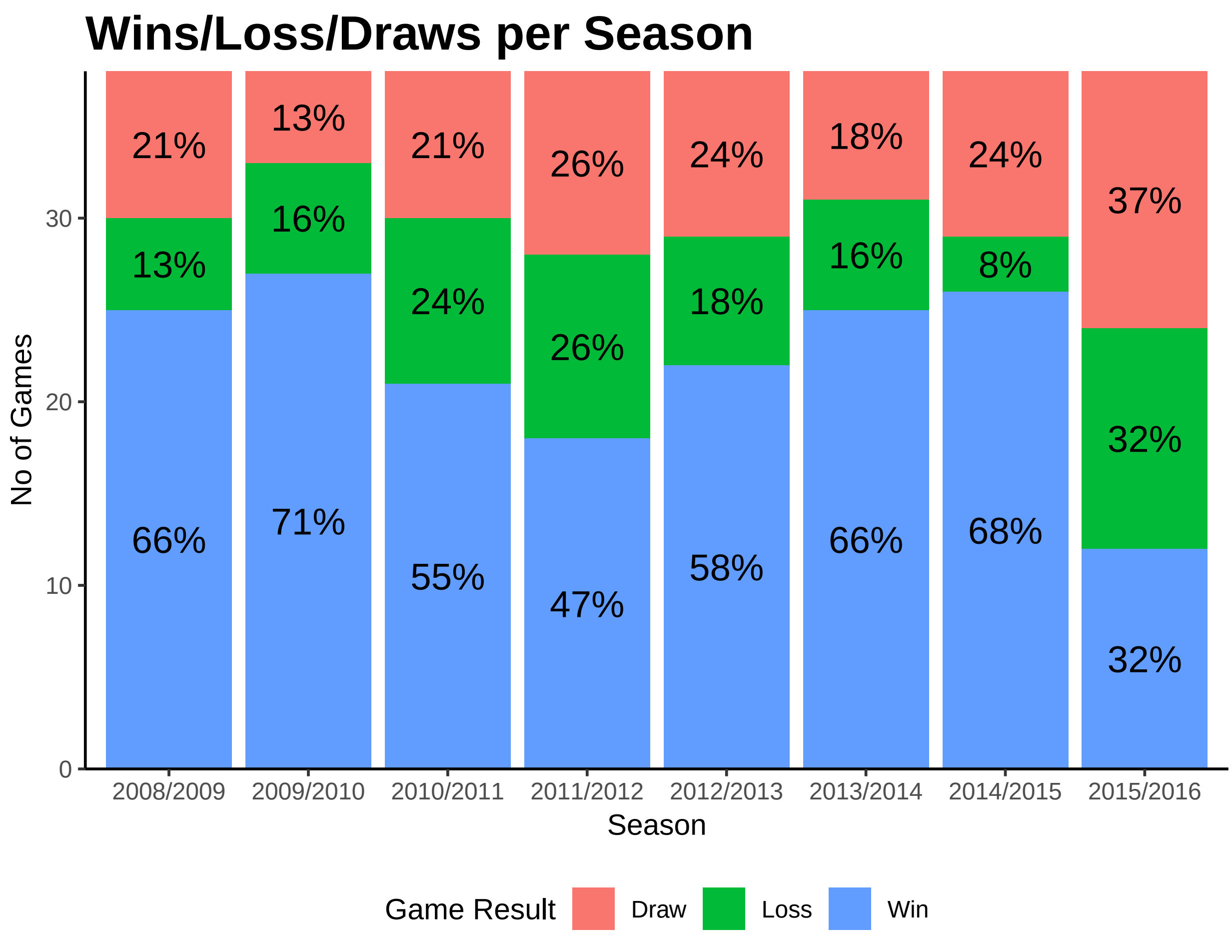


From this we see that Chelsea's performance has been more or less consistent with the exception of the last season where Chelsea was ranked 10th. This is of great concern and we hope our further analysis will help use understand some of the reasons of this poor performance in the last year. We want to deep dive further to understand the Win/Loss rate ever season.

```
matches_che <- subset(match, match$home_team_api_id == 8455)
matches_che$result<-case_when(matches_che$home_team_goal>matches_che$away_team_goal~"Win",
matches_che$home_team_goal<matches_che$away_team_goal~"Loss",
matches_che$home_team_goal==(matches_che$away_team_goal)~"Draw" )
match_player<-select(matches_che,match_api_id ,season,num_range("home_player_", 1:11),
result)
match_player_long<-gather(match_player,playerno,player_id,-c(match_api_id,result,season))
match_player_name<-merge(match_player_long,player,by.x="player_id",by.y="player_api_id",
all.x = TRUE)

matches_che_a <- subset(match, match$away_team_api_id == 8455)
matches_che_a$result<-case_when(
  matches_che_a$home_team_goal>matches_che_a$away_team_goal~"Loss",
matches_che_a$home_team_goal<matches_che_a$away_team_goal~"Win",
matches_che_a$home_team_goal==matches_che_a$away_team_goal~"Draw" )
match_player_a<-select(matches_che_a,match_api_id,season,num_range("away_player_",1:11),
result)
match_player_long_a<-gather(match_player_a, playerno, player_id, -c(match_api_id, result,
```

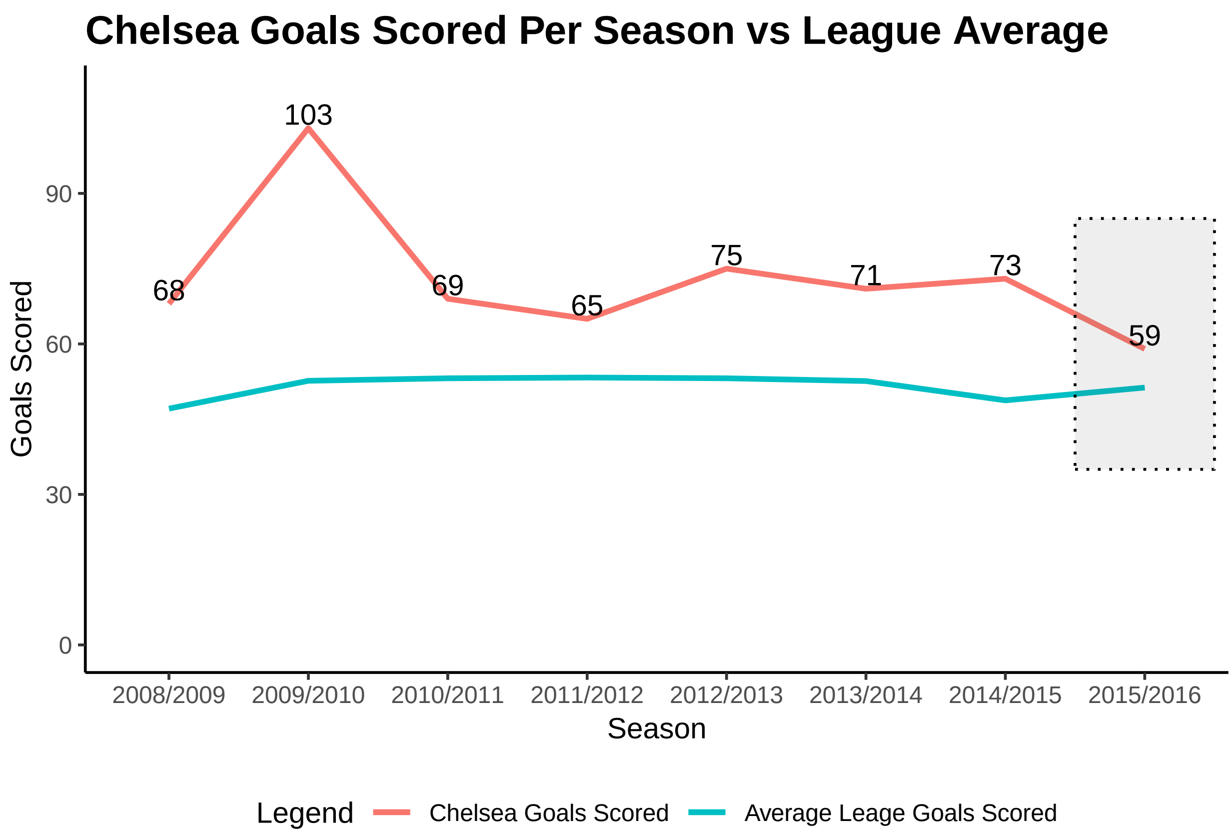
```r
season))
match_player_name_a<-merge(match_player_long_a,player,by.x="player_id",
by.y="player_api_id",all.x = TRUE)
home_away = rbind(match_player_name_a,match_player_name)

#Player Frequency Table
player_freq=home_away%>%group_by(player_name,season)%>%summarise(games_played=n())%>%
  arrange(games_played)
player_freq_overall = home_away%>%group_by(player_name)%>%summarise(games_played=n())%>%
  arrange(games_played)

matches_current_season=subset(home_away,home_away$season=='2015/2016')%>%
  select(player_name,season)
matches_current_season = matches_current_season[!duplicated(matches_current_season),]

player_freq_overall =merge(player_freq_overall,matches_current_season,
by.x ="player_name", by.y = "player_name" , all.x = TRUE)
player_freq_overall[is.na(player_freq_overall)] <- 0

#Player wins table
team_wins = home_away %>% select(season , result, match_api_id)
team_wins = team_wins[!duplicated(team_wins),]
team_wins_season=team_wins%>%group_by(season,result)%>%summarise(result_season=n())

player_wins=home_away%>%group_by(player_name,season,result)%>%summarise(result_n=n())
```

```r
ggplot(team_wins_season, aes( x = season, y = result_season , fill = result)) + geom_bar(
stat='identity') +Theme +labs(fill = "Game Result")+xlab("Season")+ylab("No of Games")+
ggtitle( "Wins/Loss/Draws per Season") + theme(legend.position="bottom") + geom_text(aes(
label=paste0(round(result_season*100/38),'%')),size=5,position=position_stack(vjust=0.5))+
  theme(plot.title=element_text(size=18,hjust=0,face="bold",colour="black",vjust=1))+
  scale_y_continuous(expand = c(0, 0),limits = c(0, 38))
```



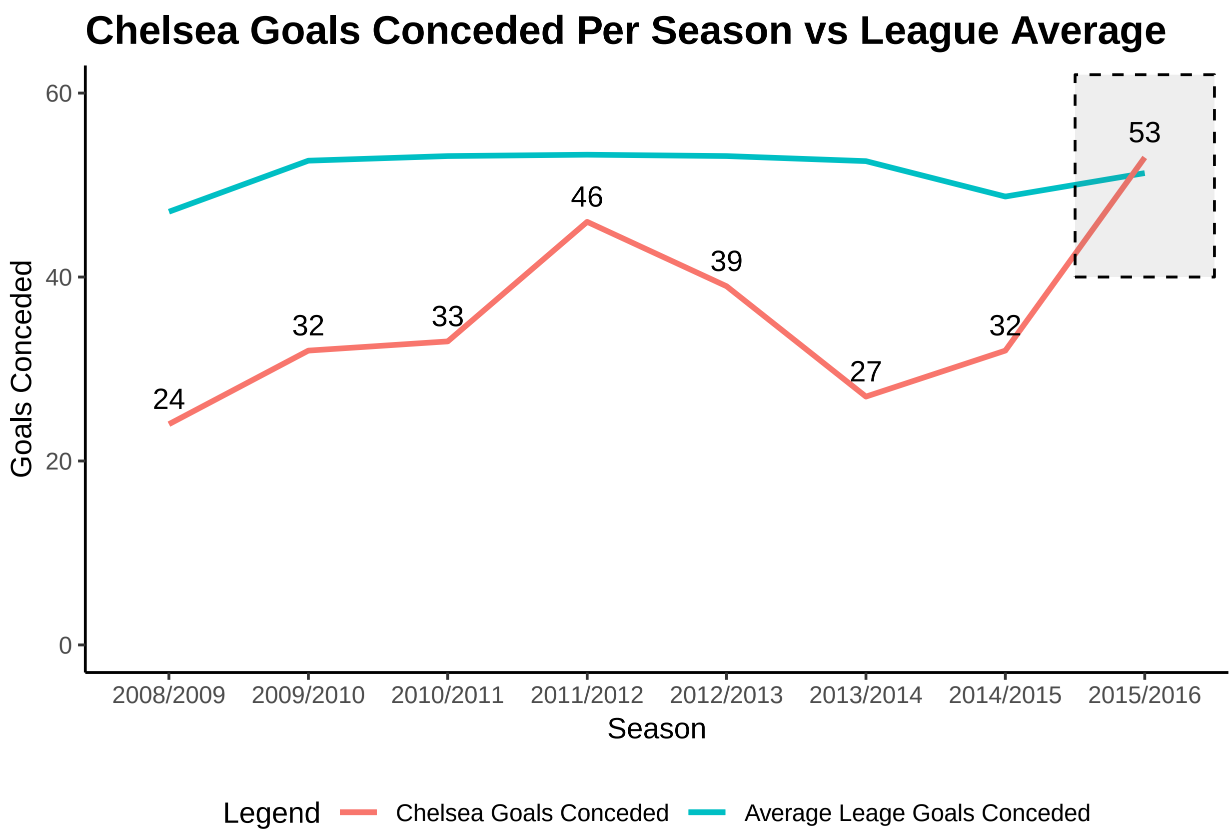
In the above graph we have the Win/Loss information. Looking at the Win Loss information we see that in the last season the % of games Chelsea won drastically fell by 50% to 12 games which is 32% of the 38 games played.

```
ggplot(league_table_final,aes ( x = season , y = total_goals_scored) )+ stat_summary(aes(
color='darblue'),fun.y='mean',size =1,geom ='line',group = 1)+coord_cartesian(ylim=c(
0, 110)) +geom_line(data =  subset(league_table_final,team_short_name=='CHE'),aes(
y= total_goals_scored,color = col1),size =1, group= 2) +Theme + annotate("rect", xmin=7.5,
xmax=8.5, ymin=35, ymax=85, color="black",linetype="dotted",  alpha=0.1)+xlab("Season") +
  ylab(
"Goals Scored")+ggtitle("Chelsea Goals Scored Per Season vs League Average")+geom_text(
data=subset(league_table_final,team_short_name=='CHE'),  aes(label = total_goals_scored),
nudge_y = 2.8)  +    scale_color_discrete(name="Legend",labels=c(
"Chelsea Goals Scored","Average Leage Goals Scored"))+theme(legend.position="bottom")+
scale_fill_manual(values=c("#CC9999", "#9999CC"))+
  theme(plot.title=element_text(size=15,hjust=0,face="bold",colour="black",vjust=1))
```



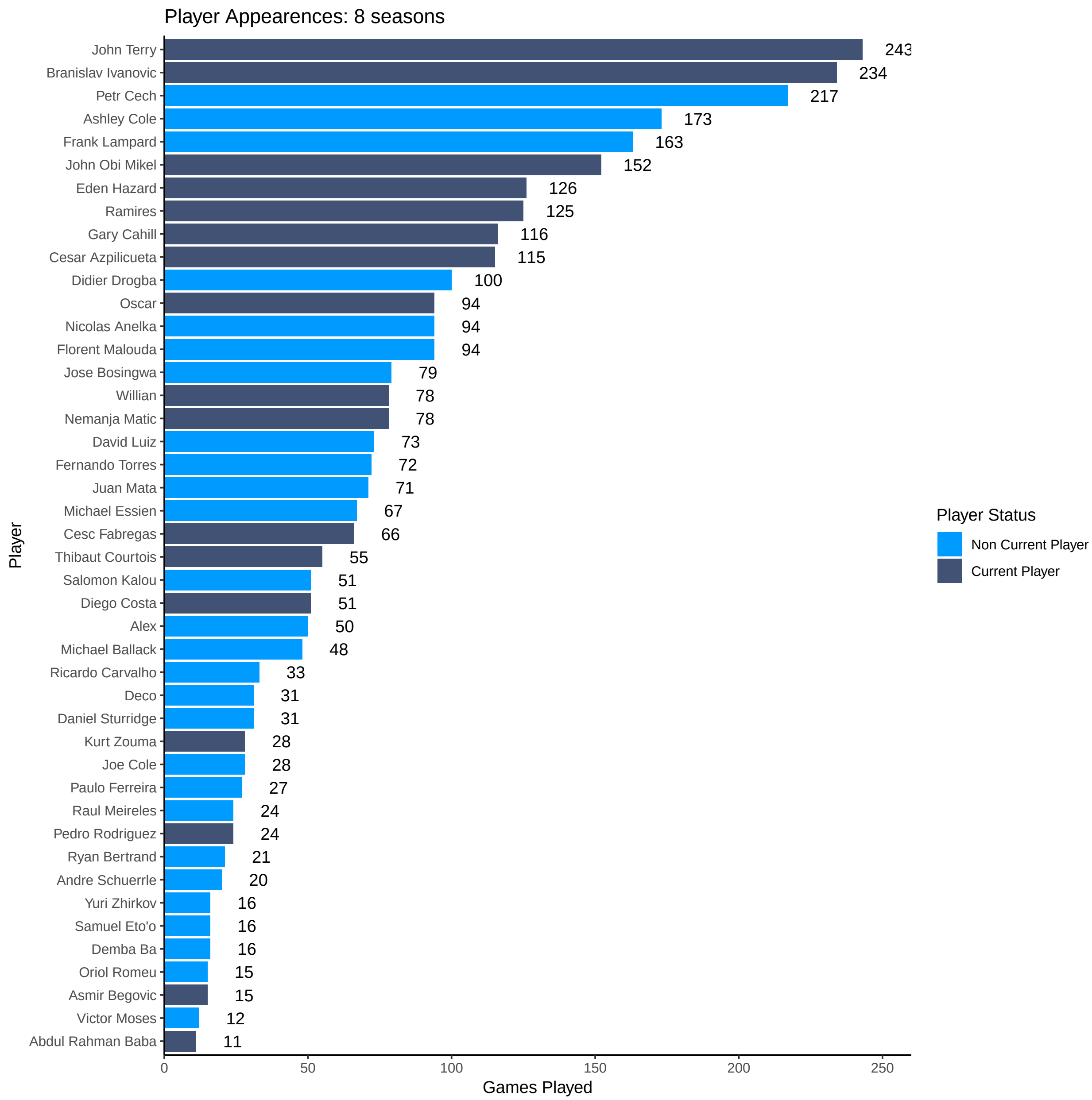
Analyzing the goals scored we see that the goals scored by Chelsea have dipped in the last year while the league average increased slightly. We will be deep diving into the performance of the strikers down the line.

```
ggplot (league_table_final, aes ( x = season , y = total_goals_conceded) )+ stat_summary(
aes(col='darblue'), fun.y = 'mean' , size =1,geom  = 'line', group = 1)+ coord_cartesian(
ylim = c(0, 60)) +geom_line(data =  subset(league_table_final,team_short_name=='CHE'),aes(
y= total_goals_conceded,color=col1),size =1,group= 2)+Theme+annotate("rect",xmin=7.5,
xmax=8.5, ymin=40, ymax=62, linetype = 'dashed',color="black", alpha=0.1)+xlab("Season")+
  ylab("Goals Conceded")+
ggtitle( "Chelsea Goals Conceded Per Season vs League Average") + geom_text(data = subset(
league_table_final,team_short_name=='CHE'),aes(label=total_goals_conceded),nudge_y=2.8)+
scale_color_discrete(name = "Legend", labels = c("Chelsea Goals Conceded",
"Average Leage Goals Conceded")) + theme(legend.position="bottom")+ scale_fill_brewer(
palette = "Dark2")+
  theme(plot.title=element_text(size=15,hjust=0,face="bold",colour="black",vjust=1))
```



Analyzing the goals conceded last season we see that the last season was the first time that **Chelsea conceded more goals than the league average!** This suggests that Chelsea's defense did not perform upto expectations and needs to be examined more closely.

```
player_freq_overall1 <- subset(player_freq_overall, player_freq_overall$games_played>=10)
ggplot(player_freq_overall1, aes(x= reorder ( player_name, games_played ),y=games_played,
fill=season))+geom_bar(stat='identity')+coord_flip()+xlab("Player")+ylab("Games Played")+
ggtitle("Player Appearences: 8 seasons ")+geom_text(aes(label=games_played),nudge_y=12.8)+
  scale_color_discrete(name = "Legend", labels = c("Chelsea Goals Conceded",
"Average Leage Goals Conceded")) +labs(fill="Player Status")+scale_fill_manual(
  "Player Status",values=c("2015/2016"="#425274",
"0" = "#009BFF"),labels = c("Non Current Player", "Current Player"))+
theme(plot.title=element_text(size=20,hjust=0,face="bold",colour="black",vjust=1))+
  Theme+scale_y_continuous(expand = c(0, 0),limits = c(0, 260))
```



From this graph we see that seven of the top 10 players by appearances still play for Chelsea. This shows that our squad is experienced with 7 players playing more than 100 matches for Chelsea.

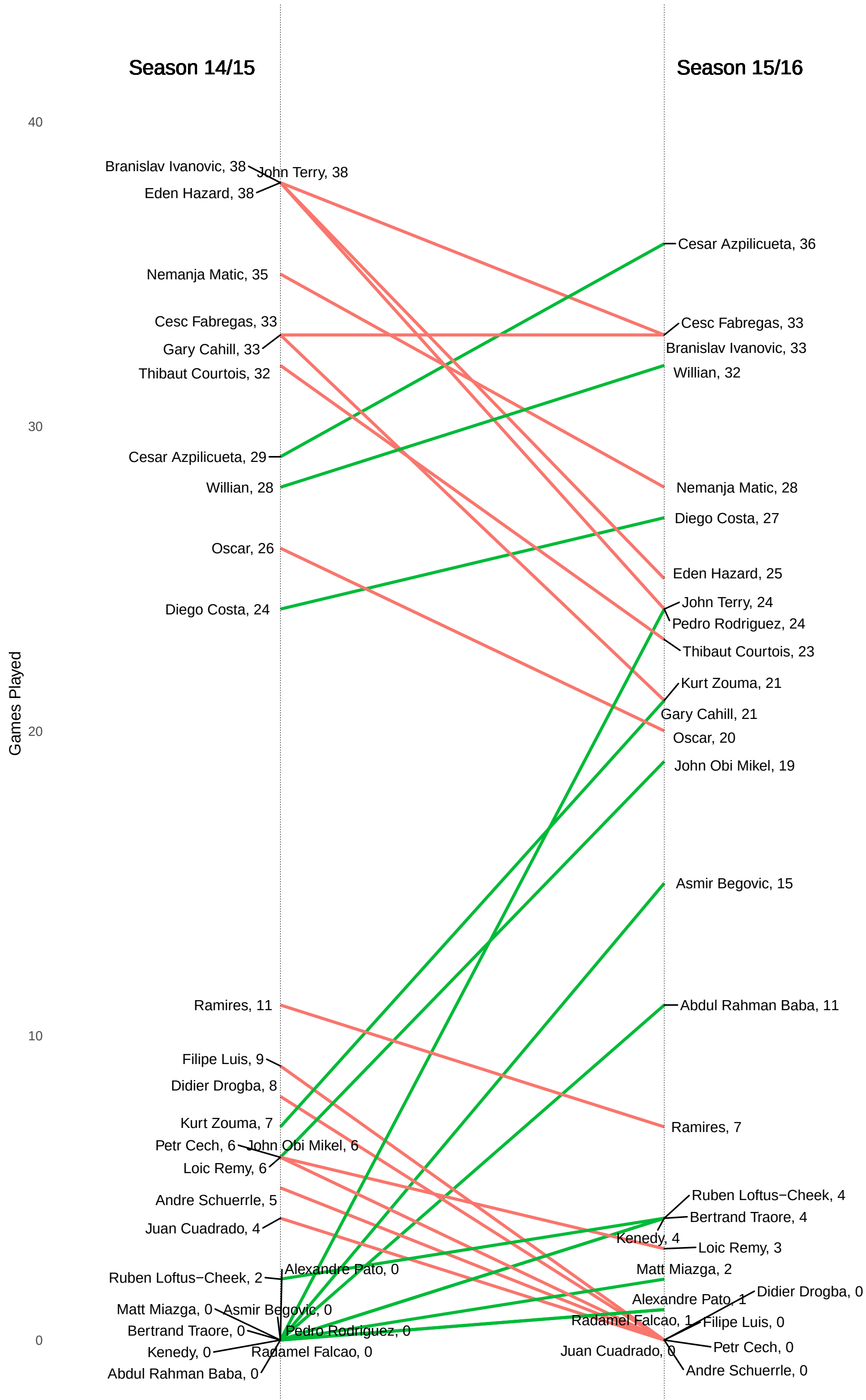
We now compare the number of games played by each player across both the seasons.

```
player_freq_current=subset(player_freq,player_freq$season =='2015/2016' |
                             player_freq$season=='2014/2015')
player_freq_current_plot = spread(player_freq_current,season, games_played)
player_freq_current_plot[is.na(player_freq_current_plot)] <- 0
left_label <- paste(player_freq_current_plot$player_name, round(
  player_freq_current_plot$`2014/2015`),sep=", ")
right_label <- paste(player_freq_current_plot$player_name, round(
  player_freq_current_plot$`2015/2016`),sep=", ")
player_freq_current_plot$class <- ifelse((player_freq_current_plot$`2014/2015`-
player_freq_current_plot$`2015/2016`) < 0, "green", "red")
```

```
ggplot(player_freq_current_plot) + geom_segment(aes(x=1, xend=2, y= `2014/2015` ,
yend=`2015/2016`, col=class), size=1, show.legend=F) +
geom_vline(xintercept=1, linetype="dashed", size=.1) + geom_vline(xintercept=2,
linetype="dashed", size=.1) +  scale_color_manual(labels = c("Up", "Down"),
values = c("green"="#00ba38", "red"="#f8766d")) +   labs(x="", y="Games Played") +
  xlim(.5, 2.5) + ylim(0,(1.1*(max(player_freq_current_plot$`2014/2015` ,
player_freq_current_plot$`2015/2016`)))) + geom_text_repel(label=left_label,
y= player_freq_current_plot$`2014/2015`, x=rep(1, NROW(player_freq_current_plot)),
hjust=1.1, size=3.5)+ geom_text_repel (label=right_label, y=
                                         player_freq_current_plot$`2015/2016`,
x=rep(2, NROW(player_freq_current_plot)), hjust=-0.1, size=3.5) + geom_text(
  label="Season 14/15",x=1, y=1.1*(max(player_freq_current_plot$`2014/2015` ,
player_freq_current_plot$`2015/2016`)),
hjust=1.2, size=5)  + geom_text(label="Season 15/16", x=2, y=1.1*(max(
player_freq_current_plot$`2014/2015` , player_freq_current_plot$`2015/2016`)), hjust=-0.1,
size=5)+theme(panel.background=element_blank(),panel.grid=element_blank(),axis.ticks =
element_blank(),axis.text.x = element_blank(),panel.border = element_blank(),
plot.margin = unit(c(1,2,1,2), "cm")) + ggtitle(
"Number of games played: 14/15 season vs 15/16 season") +labs ( subtitle =
'Players who played the most for Chelsea in 14/15 season play significately less games
in the 15/16 season')+
theme(plot.title=element_text(size=15,hjust=0,face="bold",colour="black",vjust=1))+
theme(plot.subtitle=element_text(size=12, hjust=0, face="italic", color="black"))
```

Number of games played: 14/15 season vs 15/16 season

Players who played the most for Chelsea in 14/15 season play significately less games in the 15/16 season



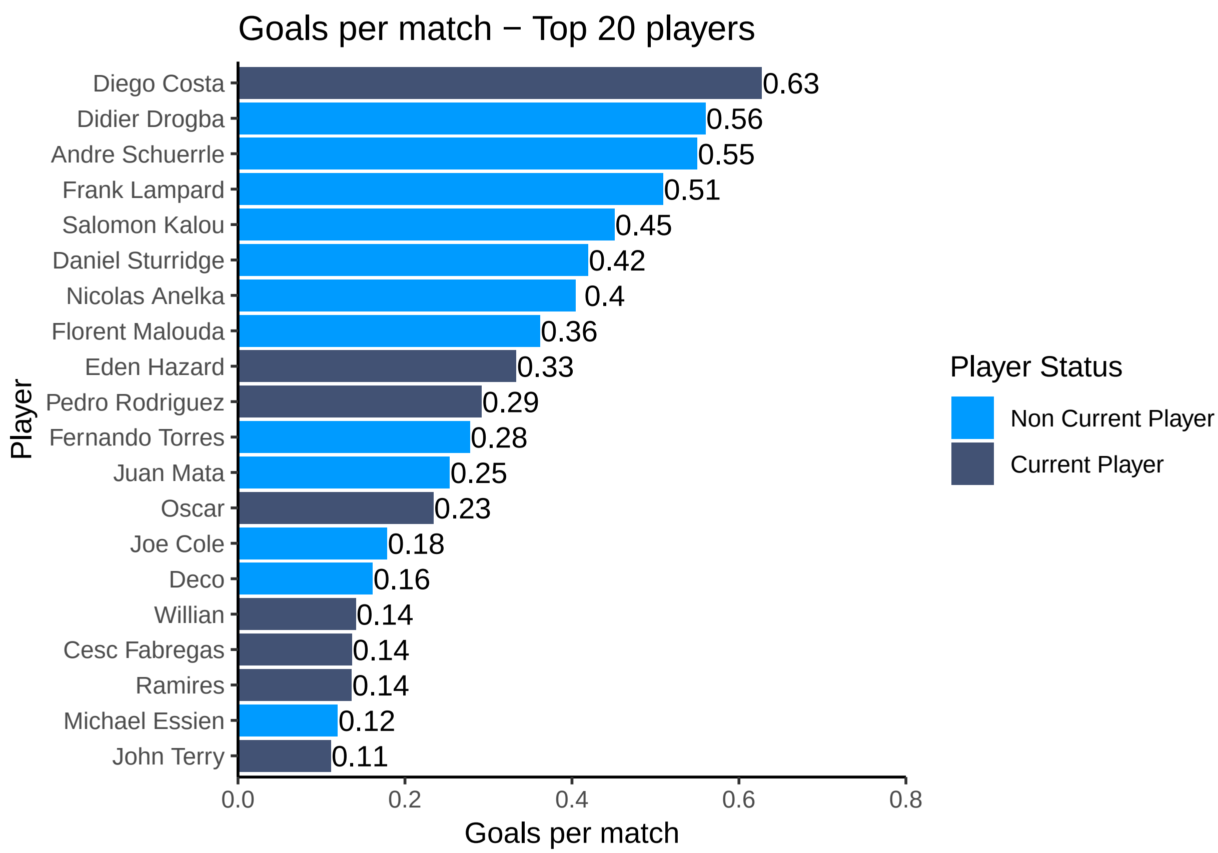
From the above graph we see a very surprising insight, the players who played the most games in the 2014/2015 season for Chelsea have played much fewer games in the 2015/2016.

In addition to the above change, it is possible that the new players are not scoring as many goals as the previous team. To analyze the contribution of players to goals and assists, we try to understand the number of goals per match by a striker and number of assists per match by a midfielder

```
## Warning: Column `team`/`team_api_id` joining factors with different levels,
## coercing to character vector

## Warning: Column `player1`/`player_api_id` joining character vector and
## factor, coercing into character vector

## Warning: Column `player2`/`player_api_id` joining character vector and
## factor, coercing into character vector
```

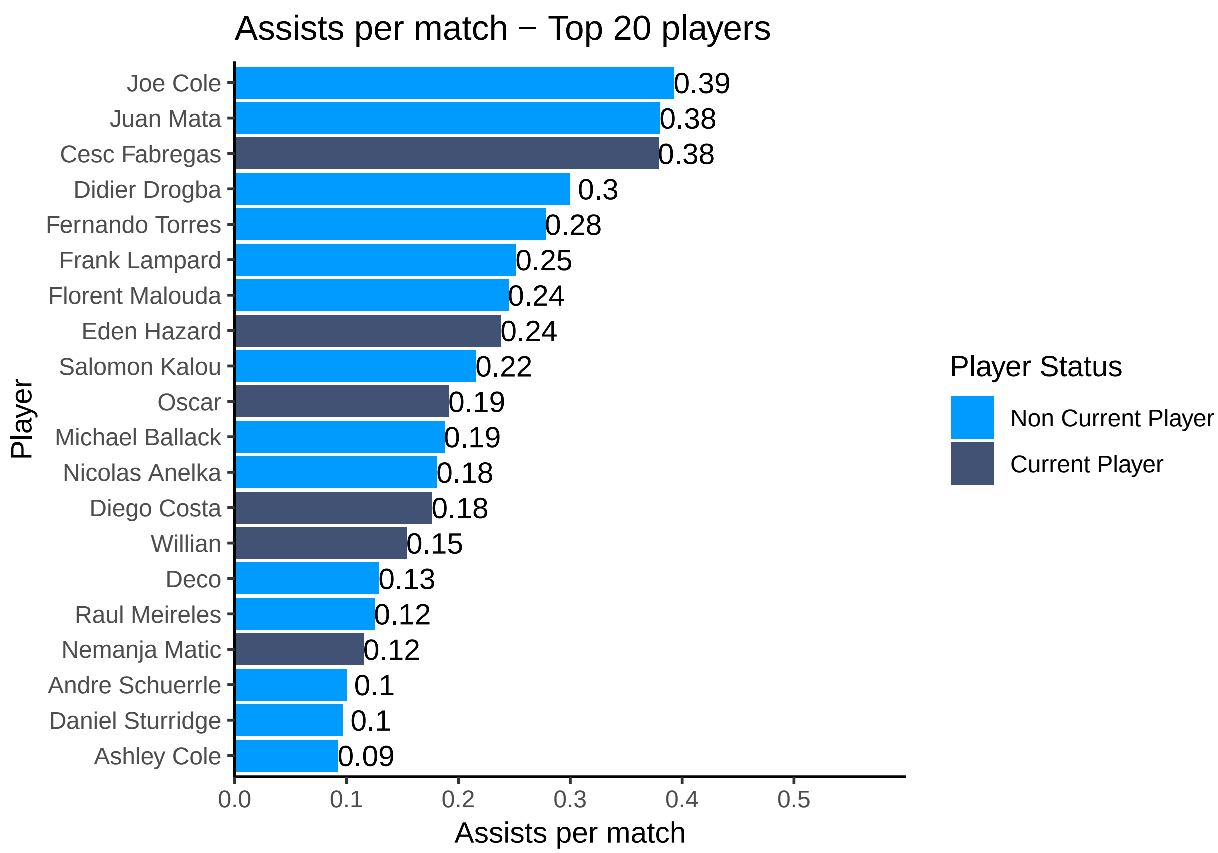
```
finaldf_non_na <- finaldf[!is.na(finaldf$lat) & !is.na(finaldf$lon), ]

chel_finaldf <- finaldf %>% filter(team_long_name == 'Chelsea', type == 'goal')

player_goals <- chel_finaldf %>% group_by(player_name.x) %>% summarize(goals = length(
  player_name.x)) %>% arrange(-goals)

player_freq <- read.csv('player_freq_overall.csv', stringsAsFactors = FALSE)

player_goals <- merge(player_goals, player_freq, by.x = 'player_name.x',
                      by.y = 'player_name', all.x = TRUE)

player_goals <- player_goals %>% mutate(goal_rate = player_goals$goals /
                                 player_goals$games_played) %>% arrange(-goal_rate)
player_goals$X <- NULL

player_goals_sub <- player_goals %>% filter(
  games_played > 16 & !is.na(goal_rate)) %>% arrange(-goal_rate) %>% head(20)
```

```
ggplot(player_goals_sub, aes(x= reorder(player_name.x, goal_rate) , y=goal_rate, fill =
                              season)) +   geom_bar(stat='identity') + coord_flip() +
  xlab("Player") + ylab("Goals per match") +
  ggtitle("Goals per match - Top 20 players") + geom_text(aes(label= round(goal_rate, 2)),
                                                         nudge_y=0.035) +
  scale_color_discrete(name = "Legend", labels = c("Chelsea Goals Conceded",
                                                  "Average Leage Goals Conceded")) +
  labs(fill="Player Status") +   theme(plot.title = element_text(hjust = 0)) +
  scale_fill_manual("Player Status",values=c("2015/2016"="#425274", "0" = "#009BFF"),
                    labels = c("Non Current Player", "Current Player"))+Theme+
  scale_y_continuous(expand = c(0, 0),limits = c(0, .8))
```




```
player_assists=chel_finaldf %>% filter(!is.na(player_name.y))%>%group_by(player_name.y)%>%
  summarize(assists = length(player_name.y)) %>% arrange(-assists)

player_assists <- merge(player_assists, player_freq, by.x = 'player_name.y',
                        by.y = 'player_name', all.x = TRUE)

player_assists <- player_assists %>% mutate(assist_rate =
                                              player_assists$assists /
                                              player_assists$games_played) %>% arrange(
                                                -assist_rate)
player_assists$X <- NULL
player_assists_sub <- player_assists %>% filter(games_played >
                                                  16 & !is.na(assist_rate)) %>%
  arrange(-assist_rate) %>% head(20)
```

```
ggplot(player_assists_sub, aes(x= reorder(player_name.y, assist_rate) , y=assist_rate,
fill = season)) + geom_bar(stat='identity') + coord_flip() + xlab("Player") +
  ylab("Assists per match") + ggtitle("Assists per match - Top 20 players") +
  geom_text(aes(label= round(assist_rate, 2)),nudge_y=0.025) +
  scale_color_discrete(name = "Legend", labels = c("Chelsea Goals Conceded",
  "Average Leage Goals Conceded")) + labs(fill="Player Status") +
  theme(plot.title = element_text(hjust = 0)) + scale_fill_manual("Player Status",
  values=c("2015/2016"="#425274", "0" = "#009BFF"),
  labels = c("Non Current Player", "Current Player"))+Theme+
  scale_y_continuous(expand = c(0, 0),limits = c(0, .6))
```



As the first step we are filtering the Match attributes table for Chelsea for home and away matches.

```r
all_matches_last2 <- all_matches %>% mutate(loc = ifelse(home_team_api_id == 8455,
  "home", "away")) %>% filter(season %in% c('2014/2015', '2015/2016'))

a <- table(all_matches_last2$result, all_matches_last2$loc)

chisq.test(a)
```

```
##
##  Pearson's Chi-squared test
##
## data:  a
## X-squared = 2.1632, df = 2, p-value = 0.339
```

Based on the chi-squared test, we can conclude that the null hypothesis of no difference between the game result with respect the location of play(home/away) cannot be rejected. Since we do not have enough evidence to support the hypothesis that the performance of the team varies with location, we are keeping our analysis location agnostic.

Diego Costa, the current primary striker, has the highest number of goals per match. But he is not supported enough by his midfielders, Fabregas/Hazard/Ramires, in terms of the goals scored per match, compared to the assists provided by older midfielders like Joe Cole or Juan Mata.

C. Association Rules

Rationale

We see that there was a huge performance gap in the last two seasons and that there was a significant number of transfers and injuries between these two seasons and players changes drastically from season 2014/2015 to season 2015/2016. We would like to understand if certain combinations of players led to a performance gap between these two seasons.

1. Does player combinations impact match result?

Comparing 2014 and 2015 seasons to understand the impact of player combinations

Analysis

Our first aim was to analyze the player combinations and match result for both seasons individually. We used the apriori algorithm for frequent itemset mining and association learning to derive player associations. We consider both the home and away matches that Chelsea played in these seasons. We used market basket analysis and rules were created for player combinations and result considering a match as a transaction to understand what combinations of players lead to a win or lose on the RHS.

Read here for in-depth understanding of market basket

Defining metrics used in the analysis

Support: The percentage of transactions that contain all of the items. The higher the support the more frequently the row occurs.

Confidence: The probability that a transaction that contains the items on the left-hand side of the rule also contains the item on the right-hand side.

Lift: lift is the ratio of confidence to expected confidence. It is a measure of how often does LHS appear with RHS, compared to what chance would predict

Source for the definitions (https://bicorner.com/2015/07/22/what-the-heck-are-association-rules-in-analytics/)

1.1 Which player combinations worked in 2014?

Data preparation for Association rules

Step 1 - Creating a subset for Home matches

```
matches_che <- subset(match, match$home_team_api_id == 8455 & (season == '2014/2015' ))
matches_che$result<-case_when(matches_che$home_team_goal>matches_che$away_team_goal~"Win",
matches_che$home_team_goal<matches_che$away_team_goal~"Loss",
matches_che$home_team_goal==matches_che$away_team_goal~"Draw" )
match_player<-select(matches_che,match_api_id ,num_range("home_player_", 1:11), result)
match_player_long<-gather(match_player, playerno, player_id, -c(match_api_id))
match_player_name<-merge(match_player_long,player,by.x = "player_id",
by.y = "player_api_id" , all.x = TRUE)
match_player_name$arules_input = case_when( !is.na(match_player_name$player_name
           ) ~ match_player_name$player_name , is.na(match_player_name$player_name
                                                 ) ~ match_player_name$player_id)
```

Step 2. Creating a subset for Away matches

```
matches_che_a <- subset(match, match$away_team_api_id == 8455 & (season == '2014/2015'))
matches_che_a$result<-case_when(matches_che_a$home_team_goal>
                                  matches_che_a$away_team_goal~"Loss",
matches_che_a$home_team_goal<matches_che_a$away_team_goal~"Win",
matches_che_a$home_team_goal==matches_che_a$away_team_goal~"Draw" )
match_player_a=select(matches_che_a,match_api_id,num_range("away_player_", 1:11),result)
match_player_long_a<-gather(match_player_a, playerno, player_id, -c(match_api_id))
match_player_name_a<-merge(match_player_long_a,player,by.x = "player_id",
```

```
by.y = "player_api_id" , all.x = TRUE)
match_player_name_a$arules_input = case_when( !is.na(match_player_name_a$player_name
    ) ~ match_player_name_a$player_name , is.na(match_player_name_a$player_name
                                                    ) ~ match_player_name_a$player_id)
```

Step 3. Combining Home and away matches

```
home_away = rbind(match_player_name_a,match_player_name)
```

Step 4. Association rules for home and away matches for 2014

Analysis of wins

We analyzed the rules which had a win on the RHS to understand what combinations were most impactful in 2014.

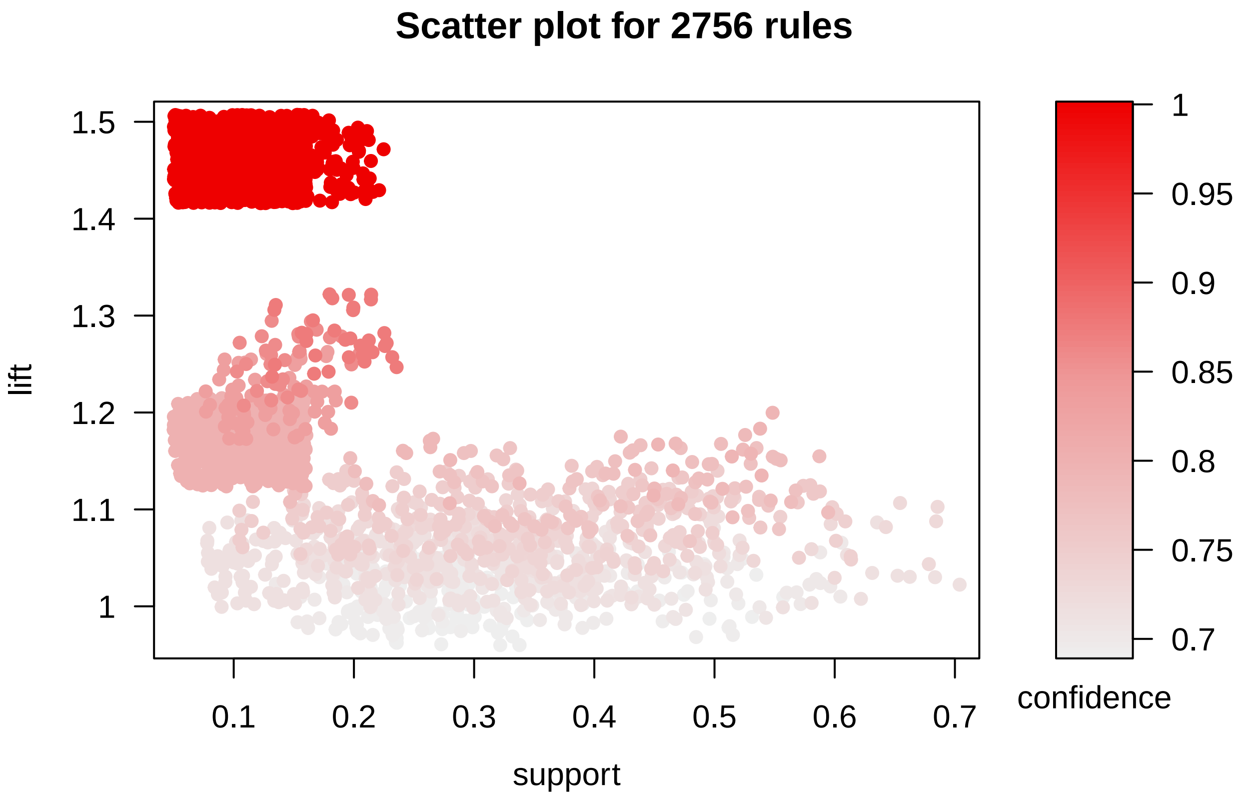
```
trans <- as(split(home_away[,"arules_input"], home_away[,"match_api_id"]), "transactions")
rules <- apriori(trans,parameter=list(supp = 0.1 , conf=0.1))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  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: 3
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[24 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [22 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10
```

```
## Warning in apriori(trans, parameter = list(supp = 0.1, conf = 0.1)): Mining
## stopped (maxlen reached). Only patterns up to a length of 10 returned!
```

```
##  done [0.00s].
## writing ... [60344 rule(s)] done [0.02s].
## creating S4 object  ... done [0.05s].
```

```
rules_win <- subset (rules, subset = (rhs %pin% "Win") & (lift>1) )
rules_win_loss_draw <- subset (rules, subset =( (rhs %pin% "Win")|(rhs %pin% "Loss")|(
  rhs %pin% "Draw")) & (lift>1) )
rule_win_df = data.frame(
       lhs = labels(lhs(rules_win)),
       rhs = labels(rhs(rules_win)),
       rules_win@quality)
rule_win_df = rule_win_df[with(rule_win_df, order(-rule_win_df$lift,
                                                     -rule_win_df$support)), ]
plot(rules_win, measure=c("support", "lift"), shading="confidence", jitter = 5)
```



Interpretation

We analyzed the rules which had a win on the RHS to understand what combinations had the highest lift. Highest lift obtained was 1.46, highest support was 0.65 and the highest confidence was 1. But we observed that there were 1109 rules that had the same confidence and lift. Added to the fact that Chelsea won in that season, we believe these were too many rules to provide any substantial insight.

We hence used the grouped matrix plot** of rules to analyze the rules clusters that were interesting.

**Grouped matrix-based visualization (Hahsler2016). Antecedents (columns) in the matrix are grouped using clustering. Groups are represented by the most interesting item (highest ratio of support in the group to support in all rules) in the group. Balloons in the matrix are used to represent with what consequent the antecedents are connected.

```
plot(rules_win, method="grouped", measure="support")
```

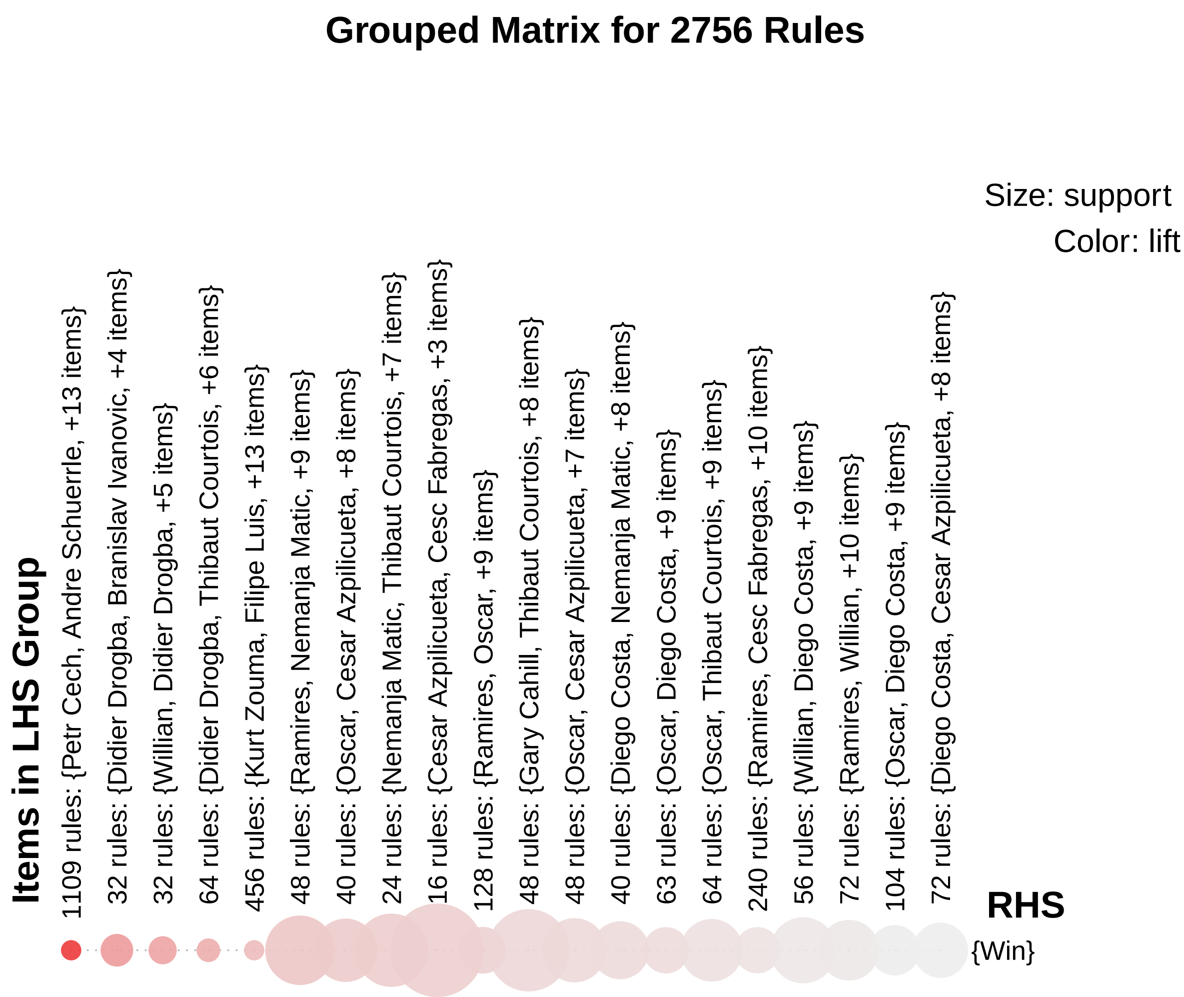


Figure : The size of the bubbles represents support and the intensity of the color represent lift

Interpretation

We see that groups with high lift included players Petr Cech and Dridier Drogba. Both these players, Petr Cech (goalkeeper) and Drogba (striker) left Chelsea after the 2014 season. Hence, our next step was to understand which players replaced the striker and goalkeeper in 2015 and how did their combination with other players work out.

1.2 What changed in 2015?

Data preparation for Association rules

Step 1 - Creating a subset for Home matches

```
## Chelase Team ID is 8455

matches_che_2015 <- subset(match, match$home_team_api_id == 8455 &
                                  (season == '2015/2016'))
```

```r
matches_che_2015$result<-case_when(matches_che_2015$home_team_goal>
matches_che_2015$away_team_goal~"Win",
matches_che_2015$home_team_goal<matches_che_2015$away_team_goal~"Loss",
matches_che_2015$home_team_goal==matches_che_2015$away_team_goal~"Draw" )

match_player_2015<-select(matches_che_2015,match_api_id ,
                          num_range("home_player_", 1:11), result)
match_player_long_2015<-gather(match_player_2015, playerno, player_id, -c(match_api_id))

match_player_name_2015<-merge(match_player_long_2015,player,by.x = "player_id",
by.y = "player_api_id" , all.x = TRUE)
match_player_name_2015$arules_input = case_when( !is.na(match_player_name_2015$player_name
  ) ~ match_player_name_2015$player_name , is.na(match_player_name_2015$player_name
    ) ~ match_player_name_2015$player_id)
```

Step 2. Creating a subset for Away matches

```r
matches_che_a_2015 <- subset(match, match$away_team_api_id == 8455 &
                               (season == '2015/2016'))
matches_che_a_2015$result<-case_when(matches_che_a_2015$home_team_goal>
matches_che_a_2015$away_team_goal~"Loss",
matches_che_a_2015$home_team_goal<matches_che_a_2015$away_team_goal~"Win",
matches_che_a_2015$home_team_goal==matches_che_a_2015$away_team_goal~"Draw" )
match_player_a_2015<-select(matches_che_a_2015,match_api_id ,
                            num_range("away_player_", 1:11), result)
match_player_long_a_2015<-gather(match_player_a_2015, playerno, player_id,
                                 -c(match_api_id))
match_player_name_a_2015<-merge(match_player_long_a_2015,player,by.x = "player_id",
                                by.y = "player_api_id" , all.x = TRUE)
match_player_name_a_2015$arules_input =
  case_when(!is.na(match_player_name_a_2015$player_name) ~
  match_player_name_a_2015$player_name , is.na(match_player_name_a_2015$player_name)
  ~ match_player_name_a_2015$player_id)
```

Step 3. Combining Home and away matches

```r
home_away_2015 = rbind(match_player_name_a_2015,match_player_name_2015)
```

Step 4. Association rules for home and away matches for 2014

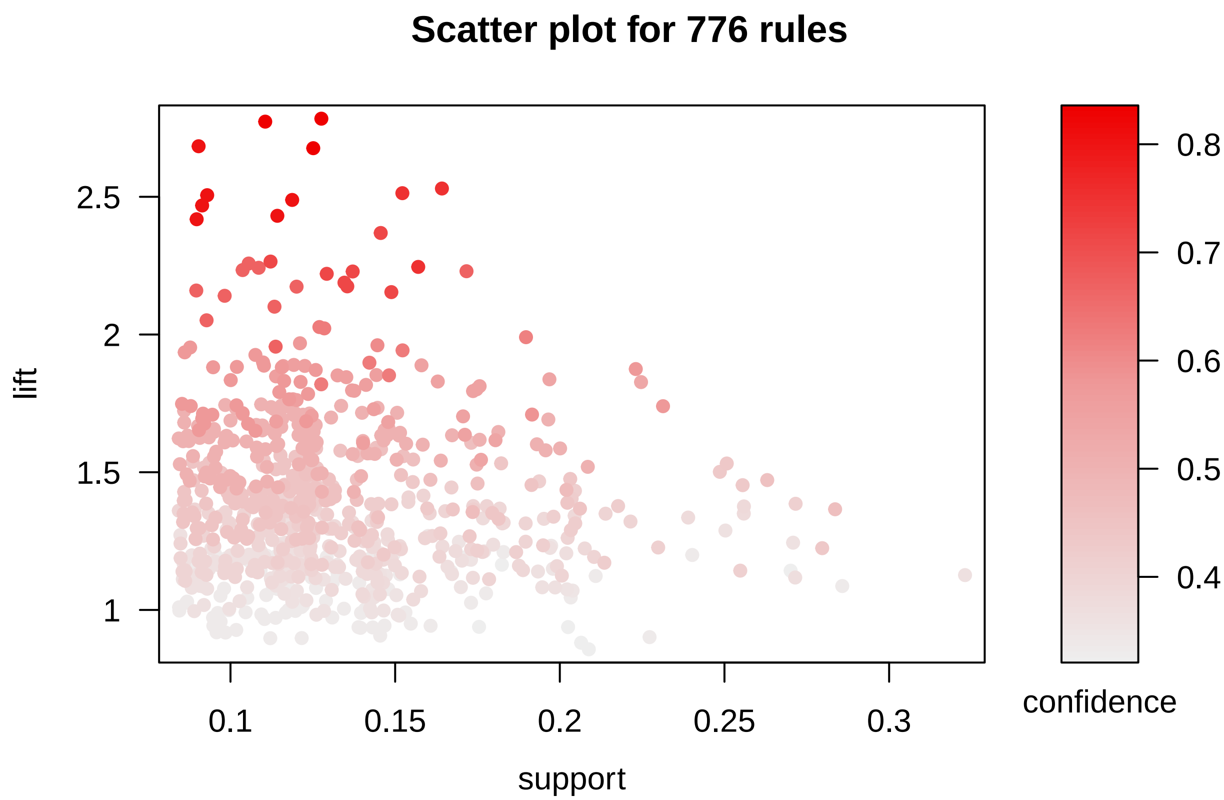
Analysis of wins

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  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: 3
##
```

```
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[27 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [23 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10
```

```
## Warning in apriori(trans_2015, parameter = list(supp = 0.1, conf = 0.1)):
## Mining stopped (maxlen reached). Only patterns up to a length of 10
## returned!
```

```
##  done [0.00s].
## writing ... [44392 rule(s)] done [0.01s].
## creating S4 object  ... done [0.02s].
```




```
plot(rules_win_loss_draw_2015, method="grouped", measure="support",
main = "Grouped matrix plot for 2015 Wins, Losses and Draws")
```

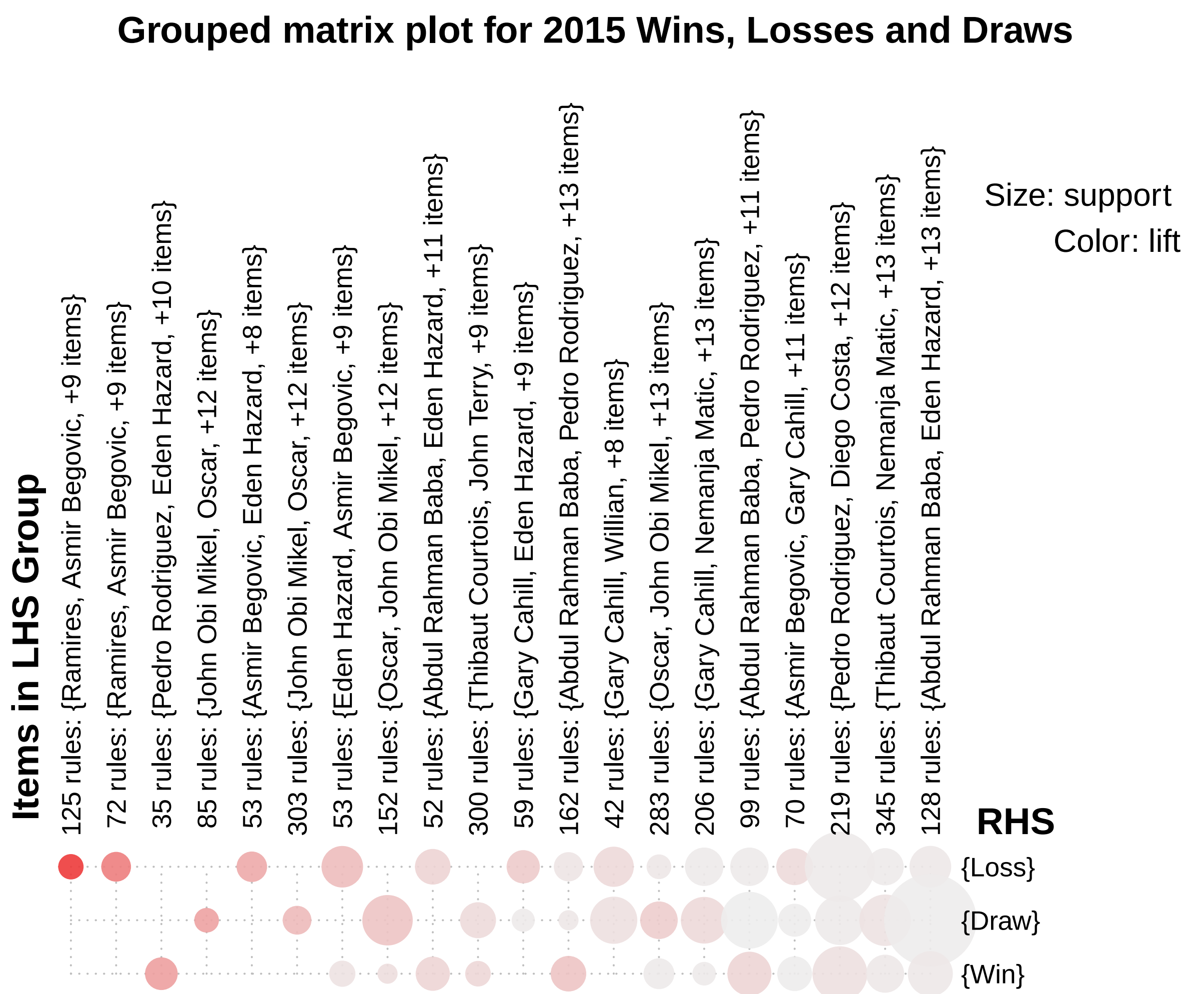


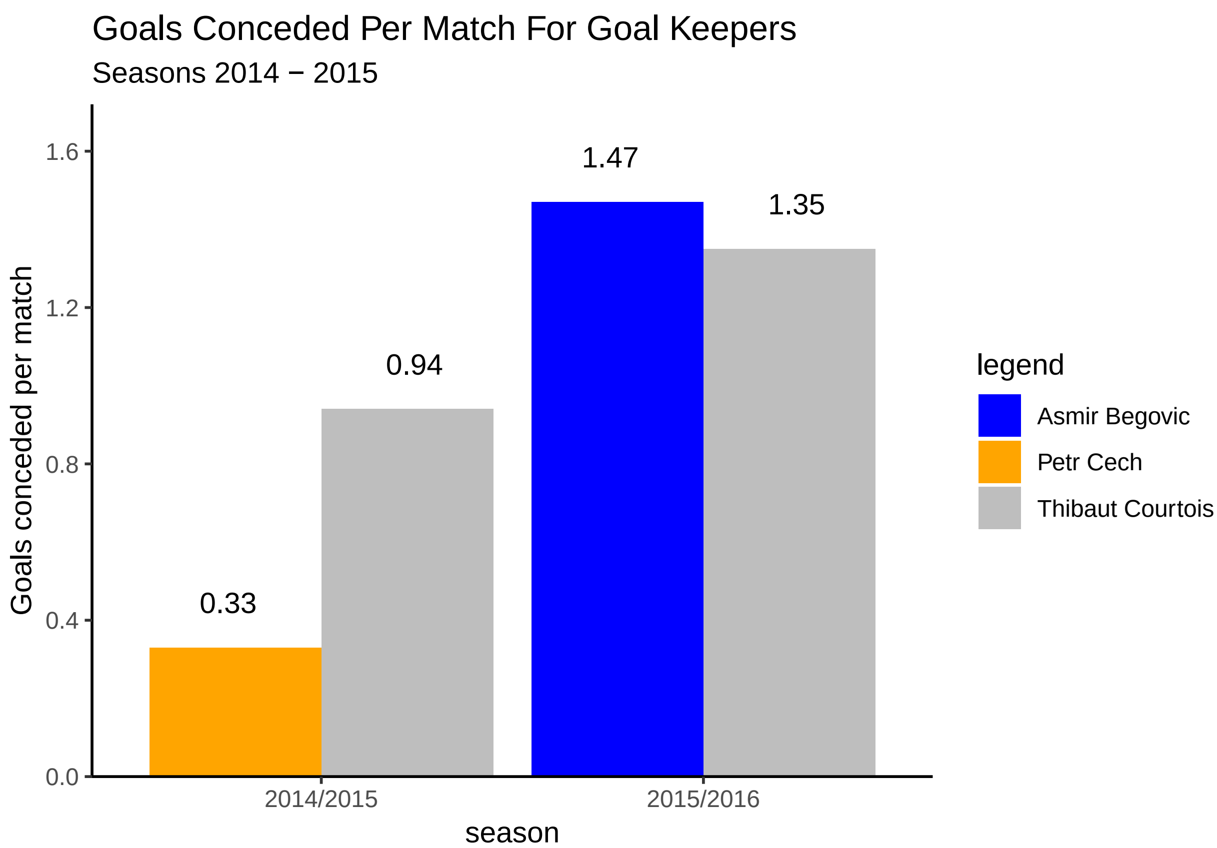
Figure : The size of the bubbles represents support and the intensity of the color represent lift

Interpretation

The above matrix plot shows that the combinations that contained Ramires and Asmir Begovic have the highest lift when looking at a 'Loss' in the RHS. We see that there is a possibility that the change in goalkeeper from Petr Cech to Asmir Begovic might have had an impact in the performance as the goals conceded was a major contributor to degraded performance. We hence analyzed the goals conceded per match for each of the goalkeepers to understand whether it was indeed the replacement that had an impact on the goals conceded.

```
matches_g_che <- subset(match, match$home_team_api_id == 8455)
matches_g_che_a <- subset(match, match$away_team_api_id == 8455)

match_player_g<-select(matches_g_che,season,match_api_id ,num_range("home_player_",
1:11),away_team_goal )
match_player_g_a<-select(matches_g_che_a,season,match_api_id ,num_range("away_player_",
1:11),home_team_goal )
```

```r
match_player_long_g<-gather(match_player_g, playerno, player_id, -c(match_api_id,season,
                                                                  away_team_goal))
match_player_long_g_a<-gather(match_player_g_a,playerno,player_id,-c(match_api_id,season,
home_team_goal))

player_g=subset(match_player_long_g,match_player_long_g$player_id%in%c(30859,
170323,46518))

player_g_a = subset(match_player_long_g_a, match_player_long_g_a$player_id %in% c(30859,
170323,46518))
player_g_goals = player_g %>% group_by(season, player_id ) %>% summarise(
goals_conceded = sum(away_team_goal), n = n())
player_g_goals_a = player_g_a %>% group_by(season, player_id ) %>% summarise(
goals_conceded = sum(home_team_goal), n = n())
player_g_final = rbind(player_g_goals,player_g_goals_a)
player_g_final = player_g_final %>% group_by( season, player_id ) %>% summarise(
goals_conceded = sum(goals_conceded ) , n = sum(n) )
player_g_final<-merge(player_g_final,player,by.x = "player_id", by.y = "player_api_id" ,
all.x = TRUE)
player_g_final_plot=player_g_final%>%mutate(goals_conceded_game=round(goals_conceded/n,
2)) %>%
filter ( season == '2014/2015' | season == '2015/2016')


ggplot( player_g_final_plot , aes( x =season , y = goals_conceded_game, label =
goals_conceded_game) ) + geom_bar(aes(fill = player_name),
position = "dodge", stat="identity") +
geom_text(position = position_dodge(width= 0.975), aes(y=goals_conceded_game+0.25,
fill=player_name,label=goals_conceded_game, hjust=0.5, vjust = 3))+ scale_fill_manual(
"legend",
values = c("Asmir Begovic" = "blue", "Petr Cech" = "orange", "Thibaut Courtois"="grey")) +
labs(title = "Goals Conceded Per Match For Goal Keepers",
subtitle = "Seasons 2014 - 2015", y = "Goals conceded per match")+ Theme +
scale_y_continuous(expand = c(0, 0))
```



Interpretation

Asmir Begovic in the last two seasons has conceded 1.6 goals per match while Cech conceded 0.33 goals per match. We can hence conclude that goalkeeper replacement had an impact on performance and recommend that Asmir Begovic is replaced with a keeper whose characteristics match Petr Chec.

Goals conceded can also be impacted by a weak defensive line. This led us to analyze the defender's combinations for both the seasons to understand what is the best combination for defenders and what combinations do not work.

2. Using association rules to mine for winning player combinations

2.1 What is the best defense line?

Analysis for defenders

Association mining using apriori algorithm for defenders for the season 2015. We are trying to understand the combination of defenders that leads to wins and what combinations of players that generally results in a loss.

```
#Home
matches_che <- subset(match, match$home_team_api_id == 8455 & (season == '2015/2016' ) )
matches_che$result<-case_when(matches_che$home_team_goal>matches_che$away_team_goal~"Win",
matches_che$home_team_goal<matches_che$away_team_goal~"Loss",
matches_che$home_team_goal==matches_che$away_team_goal~"Draw" )
match_player<-select(matches_che,match_api_id ,num_range("home_player_", 1:11), result)
match_player_long<-gather(match_player, playerno, player_id, -c(match_api_id))
match_player_name<-merge(match_player_long,player,by.x = "player_id",
by.y = "player_api_id" ,
all.x = TRUE)
```

```r
match_player_name = subset ( match_player_name , match_player_name$player_id %in% c(
23783,281207,30627,31306,324910,72541, 41167, 'Draw' ,'Loss' ,'Win'))
match_player_name$arules_input = case_when( !is.na(match_player_name$player_name
) ~ match_player_name$player_name , is.na(
match_player_name$player_name) ~ match_player_name$player_id)

#Away team
matches_che_a <- subset(match, match$away_team_api_id == 8455 & (season == '2015/2016'))

matches_che_a$result<-case_when(matches_che_a$home_team_goal>
                                  matches_che_a$away_team_goal~"Loss",
matches_che_a$home_team_goal<matches_che_a$away_team_goal~"Win",
matches_che_a$home_team_goal==matches_che_a$away_team_goal~"Draw" )

match_player_a <- select(matches_che_a,match_api_id ,num_range("away_player_", 1:11),
                       result)
match_player_long_a<-gather(match_player_a, playerno, player_id, -c(match_api_id))
match_player_name_a<-merge(match_player_long_a,player,
by.x = "player_id", by.y = "player_api_id" ,
all.x = TRUE)

match_player_name_a = subset ( match_player_name_a , match_player_name_a$player_id %in% c(
  23783,281207,30627,31306,324910,72541, 41167, 'Draw' ,'Loss' ,'Win'))

match_player_name_a$arules_input = case_when( !is.na(match_player_name_a$player_name
                  ) ~ match_player_name_a$player_name , is.na(
                  match_player_name_a$player_name) ~ match_player_name_a$player_id)

#Union away and home
home_away = rbind(match_player_name_a,match_player_name)
trans <- as(split(home_away[,"arules_input"], home_away[,"match_api_id"]), "transactions")

# Running Apriori algorithm
rules <- apriori(trans,parameter=list(supp = 0.1 , conf=0.1))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  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: 3
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[9 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 done [0.00s].
```

```
## writing ... [284 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

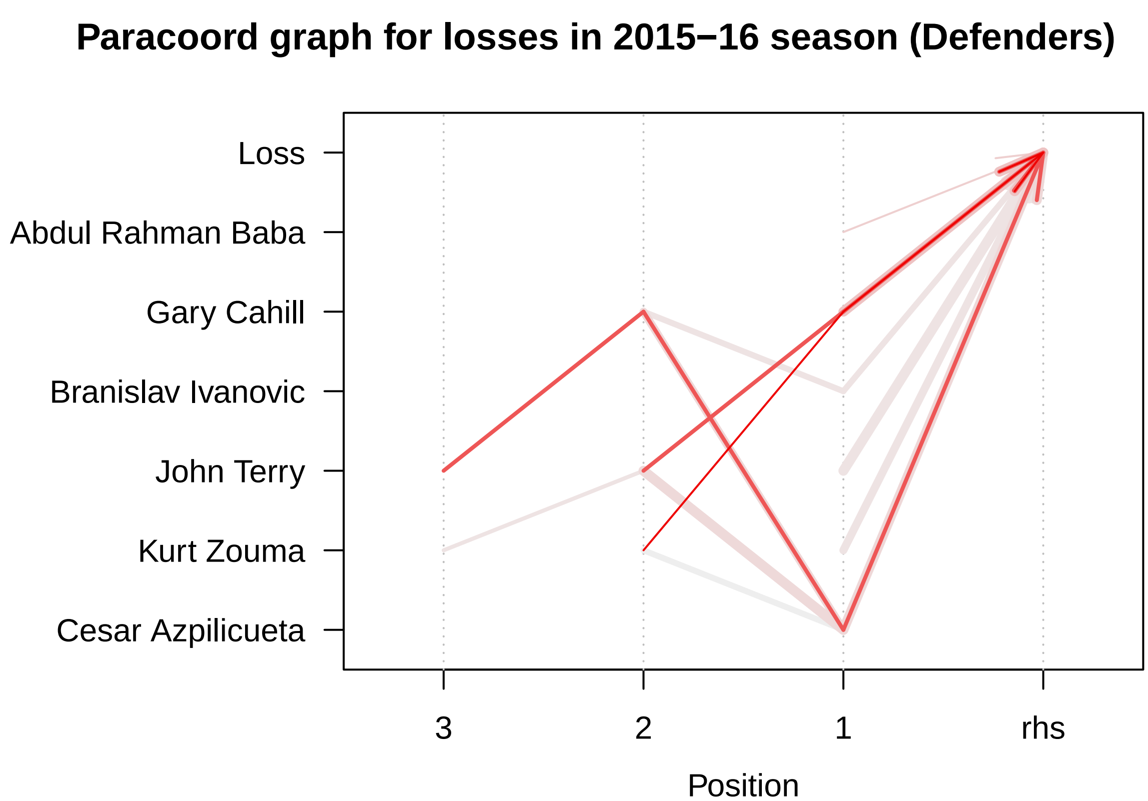
Analysis of loss

We use a paracoord graph** here to show all the combinations that show high confidence when the RHS is a loss

**Parallel coordinates plots are designed to visualize multidimensional data where each dimension is displayed separately on the x-axis and the y-axis is shared. Each data point is represented by a line connecting the values for each dimension. Parallel coordinates plots were used previously to visualize discovered classification rules (Han, An, and Cercone 2000) and association rules (Yang 2003). Yang (2003) displays the items on the y-axis as nominal values and the x-axis represents the positions in a rule, i.e., first item, second item, etc. Instead of a simple line, an arrow is used where the head points to the consequent item. Arrows only span enough positions on the x-axis to represent all the items in the rule, i.e., rules with fewer items are shorter arrows

Reading on paracoord graph

```
## Loss
loss_rules<-subset(rules, subset = rhs %pin% "Loss" & lift>=1)
plot(loss_rules,method = "paracoord",control= list(
  main = "Paracoord graph for losses in 2015-16 season (Defenders)"))
```



The width of the arrows represents support and the intensity of the color represent confidence

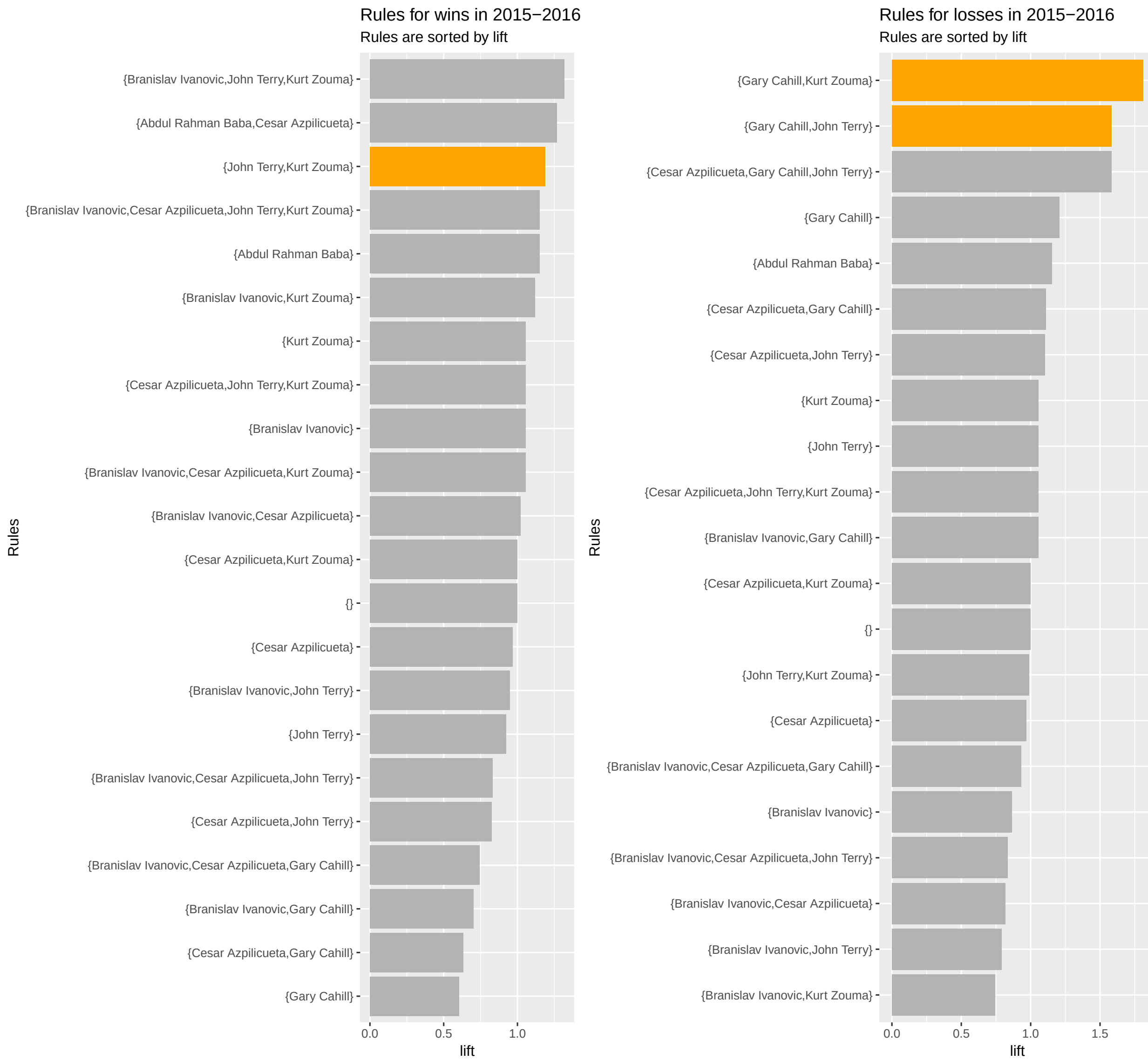
Interpretation

We see from the paracoord graph that all the combinations that show high confidence when the RHS as a loss includes Gary Cahill. And the combination of defense line shown here that leads to loss is John Terry, Cesar Azpilicueta, Gary Cahill. We hence wanted to compare the combinations that resulted in the loss to the combinations that resulted in wins

Comparing wins to loss for defense

We subset the RHS of the rules for wins and loss and sort the rules by lift.

```
#Winning
win_rules<-subset(rules, subset = rhs %pin% "Win" )
rule_win_df = data.frame(
       lhs = labels(lhs(win_rules)),
       rhs = labels(rhs(win_rules)),
       win_rules@quality)
rule_win_df<-rule_win_df[order(-rule_win_df$lift),]

rule_win_df$colors<-case_when(rule_win_df$lhs == '{John Terry,Kurt Zouma}'~"one",
rule_win_df$lhs == '{Branislav Ivanovic,Cesar Azpilicueta,John Terry,
Kurt Zouma}'~ 'two', TRUE ~ "three")

loss_rules<-subset(rules, subset = rhs %pin% "Loss" )
rule_loss_df = data.frame(
       lhs = labels(lhs(loss_rules)),
       rhs = labels(rhs(loss_rules)),
       loss_rules@quality)
rule_loss_df<-rule_loss_df[order(-rule_loss_df$lift),]

rule_loss_df$colors<-case_when(rule_loss_df$lhs == '{Gary Cahill,Kurt Zouma}'~"one",
rule_loss_df$lhs == '{Gary Cahill,John Terry}'~'two', TRUE~'three')

p1<-ggplot( rule_win_df , aes( x = reorder(lhs,lift) , y = lift, fill = colors,
 label = lift )) + geom_bar(stat="identity") + coord_flip()+
  scale_fill_manual(values = c("orange", "gray70","midnightblue")) +
theme(legend.position="None") +
labs(title = "Rules for wins in 2015-2016", y= "lift", x = "Rules",
subtitle = "Rules are sorted by lift")
p2<-ggplot( rule_loss_df , aes( x = reorder(lhs,lift) , y = lift,fill = colors,
label = lift) ) +geom_bar(stat="identity") + coord_flip()+
scale_fill_manual(values = c("orange", "gray70","orange")) +
theme(legend.position="none")+ labs(title = "Rules for losses in 2015-2016", y= "lift",
x = "Rules", subtitle = "Rules are sorted by lift")

grid.arrange(p1, p2,  ncol = 2, nrow = 1)
```



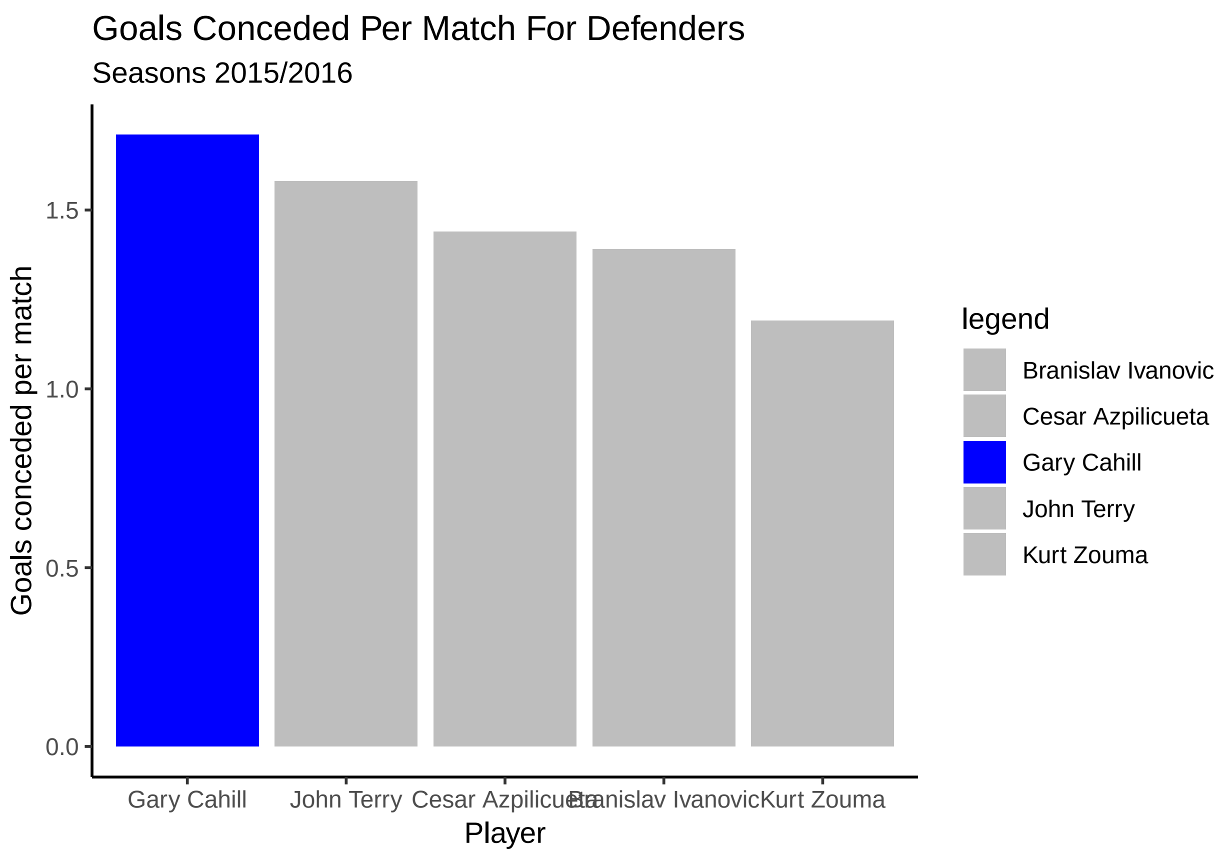
Interpretation

- The combination (in blue) Branislav Ivanovich, Cesar Azpilicueta, John Terry and Kurt Zouma as the four defenders has the highest win lift. That is, we are 1.15 times more likely to win when Branislav Ivanovich, Cesar Azpilicueta, John Terry and Kurt Zouma play as the defenders, compared to other matches. On further exploration we found that this combination has only 4 out of 38 matches in the season. It could be that using the wrong combination of players in the defense line led to a weak defense and hence higher goals conceded.
- Cahill has more than 1.5 times chance of losing matches over random chance in all combination.

we wanted to analyze how Cahill performes compared to other defenders based on goals conceded per match.

```
matches_g_che <- subset(match, match$home_team_api_id == 8455)
matches_g_che_a <- subset(match, match$away_team_api_id == 8455)
match_player_g<-select(matches_g_che,season,match_api_id ,num_range("home_player_",
                                                                  1:11),away_team_goal)
match_player_g_a<-select(matches_g_che_a,season,match_api_id ,num_range("away_player_",
1:11),home_team_goal )
```

```r
match_player_long_g=gather(match_player_g,playerno,player_id,-c(match_api_id,season,
                                                                    away_team_goal))
match_player_long_g_a<-gather(match_player_g_a, playerno, player_id, -c(match_api_id,season,
                                                                        home_team_goal))

player_g = subset(match_player_long_g, match_player_long_g$player_id %in% c(23783,30627,
31306,72541,281207))
player_g_a = subset(match_player_long_g_a, match_player_long_g_a$player_id %in% c(23783,
30627,31306,72541,281207))
player_g_goals = player_g %>% group_by(season, player_id ) %>% summarise(
goals_conceded = sum(
away_team_goal), n = n())
player_g_goals_a = player_g_a %>% group_by(season, player_id ) %>% summarise(
goals_conceded = sum(
home_team_goal), n = n())

player_g_final = rbind(player_g_goals,player_g_goals_a)

player_g_final = player_g_final %>% group_by( season, player_id ) %>% summarise(
goals_conceded = sum(
goals_conceded ) , n = sum(n) )
player_g_final<-merge(player_g_final,player,by.x = "player_id", by.y = "player_api_id" ,
all.x = TRUE)
player_d_final_plot = player_g_final %>% mutate(goals_conceded_game = round(
goals_conceded/n,2)) %>%
filter ( season == '2015/2016')
ggplot( player_d_final_plot , aes( x =reorder(player_name,goals_conceded_game ,
function(x) -(x)) , y = goals_conceded_game,
label = goals_conceded_game) ) + geom_bar(aes(fill = player_name),
position = "dodge", stat="identity")+Theme+
scale_fill_manual("legend", values = c("Gary Cahill" = "blue", "John Terry" = "grey",
"Branislav Ivanovic" = "grey",
"Kurt Zouma" = "grey","Cesar Azpilicueta" = "grey")) +
labs(title = "Goals Conceded Per Match For Defenders", subtitle = "Seasons 2015/2016",
y = "Goals conceded per match", x= "Player")
```



We see that Cahill was the worst performer from the above graph.

Poor performance in the season can also be attributed to low goals scored. We now analyze the midfielders and striker combinations to find out which team is optimal.

2.2 What is the best combination of midfielders and strikers?

Analysis for mid fielders and strikers

Association mining using apriori algorithm for midfielders and strikers for the seasons 2014 and 2015. We are trying to understand the combination that leads to wins and what combinations of players that generally results in a loss. We are using the combination of attacking midfielders and strikers. We use the cordinates to subset for the attacking midfielders.

```
#Home
matches_che <- subset(match, match$home_team_api_id == 8455 & (season == '2015/2016' ) )
matches_che$result<-case_when(matches_che$home_team_goal>matches_che$away_team_goal~"Win",
matches_che$home_team_goal<matches_che$away_team_goal~"Loss",
matches_che$home_team_goal==matches_che$away_team_goal~"Draw" )

match_player<-select(matches_che,match_api_id ,num_range("home_player_", 1:11), result)
match_player_long<-gather(match_player, playerno, player_id, -c(match_api_id))
match_player_name<-merge(match_player_long,player,by.x = "player_id",
                         by.y = "player_api_id" , all.x = TRUE)

match_player_name=subset(match_player_name,match_player_name$player_id%in%c(30613,
79574,94086,    107417, 128864, 150250, 155066, 178812, 467354, 604982, 19243,
22543,30679,    30822,  30853,  33639,  35411,  37804,  39987,  46554,  51553,
```

```r
181276, 292462,303059, 'Draw' ,'Loss' ,'Win'))
match_player_name$arules_input = case_when( !is.na(match_player_name$player_name
                      ) ~ match_player_name$player_name , is.na(
                      match_player_name$player_name) ~ match_player_name$player_id)

#Away team
matches_che_a <- subset(match, match$away_team_api_id == 8455 & (season == '2015/2016'))

matches_che_a$result<-case_when(matches_che_a$home_team_goal>
                                  matches_che_a$away_team_goal~"Loss",
matches_che_a$home_team_goal<matches_che_a$away_team_goal~"Win",
matches_che_a$home_team_goal==matches_che_a$away_team_goal~"Draw" )

match_player_a<-select(matches_che_a,match_api_id ,num_range("away_player_",1:11),result)
match_player_long_a<-gather(match_player_a, playerno, player_id,
-c(match_api_id))
match_player_name_a<-merge(match_player_long_a,player,by.x = "player_id",
by.y = "player_api_id" ,
all.x = TRUE)

match_player_name_a = subset ( match_player_name_a , match_player_name_a$player_id %in% c(
30613,
79574,94086,    107417, 128864, 150250, 155066, 178812, 467354, 604982, 19243,  22543,
30679,30822,    30853,  33639,  35411,  37804,  39987,  46554,  51553,  181276, 292462,
303059,'Draw' ,'Loss' ,'Win'))

match_player_name_a$arules_input = case_when( !is.na(match_player_name_a$player_name
) ~ match_player_name_a$player_name , is.na(match_player_name_a$player_name
) ~ match_player_name_a$player_id)

#Union away and home
home_away = rbind(match_player_name_a,match_player_name)
trans <- as(split(home_away[,"arules_input"], home_away[,"match_api_id"]),
            "transactions")

# Running Apriori algorithm
rules <- apriori(trans,parameter=list(supp = 0.1 , conf=0.1))
```

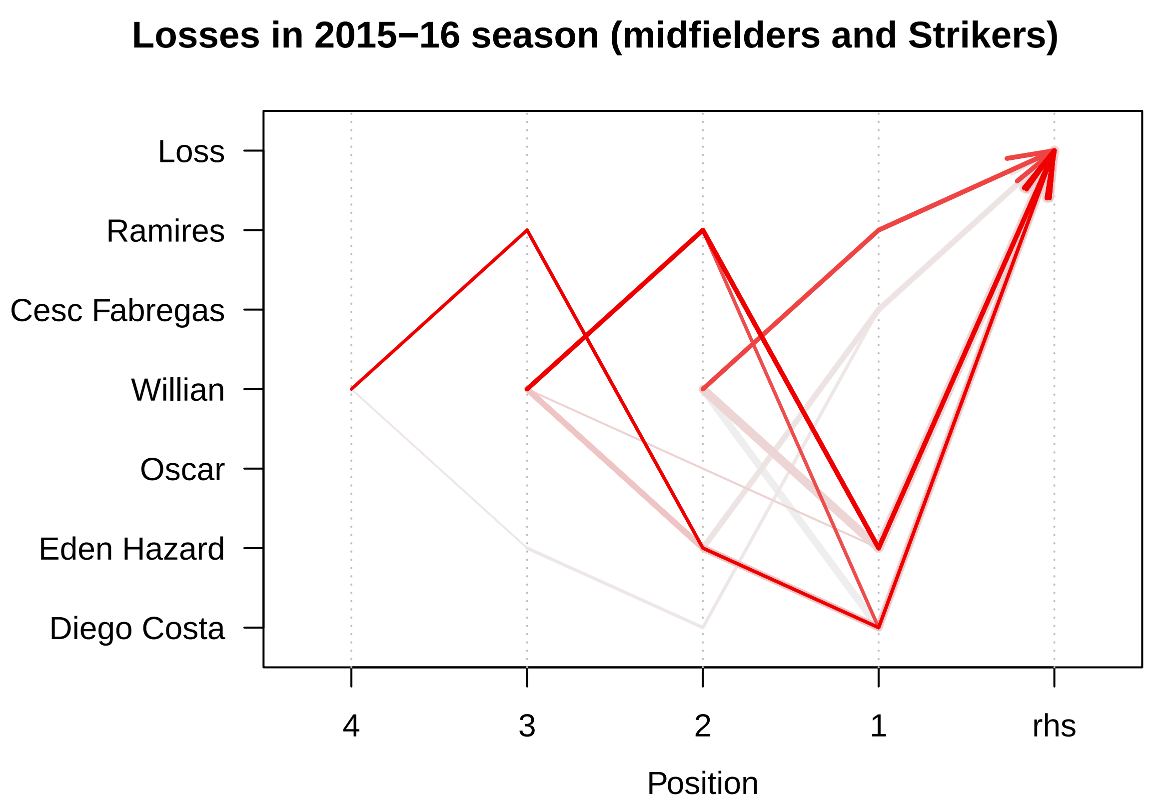
```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 3
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[16 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 done [0.00s].
## writing ... [531 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Analysis of loss

We use a paracoord graph here to show all the combinations that show high confidence when the RHS is a loss

```
## Loss
loss_rules<-subset(rules, subset = rhs %pin% "Loss" & lift>=1)
plot(loss_rules,method = "paracoord",control= list(
  main = "Losses in 2015-16 season (midfielders and Strikers)"))
```



The width of the arrows represents support and the intensity of the color represent confidence

Interpretation

We see from the paracoord graph that all the combination Ramires and Willian played has a very high confidence for loss. We also see that the combination Willian - Ramires Hazard has a high confidence for loss. We hence wanted to divide the data into win vs loss and see the difference

Comparing wins to loss

We subset the RHS of the rules for wins and loss and sort the rules by lift.

```
#Winning
win_rules<-subset(rules, subset = rhs %pin% "Win" )
rule_win_df = data.frame(
```

```r
       lhs = labels(lhs(win_rules)),
       rhs = labels(rhs(win_rules)),
       win_rules@quality)
rule_win_df<-rule_win_df[order(-rule_win_df$lift),]

rule_win_df$colors<-case_when(rule_win_df$lhs == '{Cesc Fabregas,Diego Costa,
                              Eden Hazard,Pedro Rodriguez,Willian}' ~ "one", TRUE ~ "two")

loss_rules<-subset(rules, subset = rhs %pin% "Loss" )
rule_loss_df = data.frame(
       lhs = labels(lhs(loss_rules)),
       rhs = labels(rhs(loss_rules)),
       loss_rules@quality)
rule_loss_df<-rule_loss_df[order(-rule_loss_df$lift),]

rule_loss_df$colors<-case_when(rule_loss_df$lhs == '{Diego Costa,Eden Hazard,Ramires,
Willian}'~"one", TRUE~'three')

#plot(win_rules, method="grouped", measure="support")
#plot(win_rules,method = "paracoord", measure = "lift")
p1<-ggplot( rule_win_df , aes( x = reorder(lhs,lift) ,
                               y = lift, fill = colors, label = lift )) +
  geom_bar(stat="identity") + coord_flip() +scale_fill_manual(values = c(
    "midnightblue","gray70")) + theme(legend.position="none") + labs(
      title = "Wins in 2015-2016", y= "lift", x = "Rules",
      subtitle = "Rules are sorted by lift")


p2<-ggplot( rule_loss_df , aes( x = reorder(lhs,lift) ,
                                y = lift,fill = colors, label = lift) ) +
  geom_bar(stat="identity") + coord_flip()+scale_fill_manual(values = c(
    "orange","gray70")) +  theme(legend.position="none")+ labs(
      title = "Losses in 2015-2016", y= "lift", x = "Rules",
      subtitle = "Rules are sorted by lift")

grid.arrange(p1, p2,  ncol = 2, nrow = 1)
```

Interpretation

We see that (in blue) Diego costa as the striker with Cesc Fabregas,Eden Hazard,Pedro Rodriguez or Willian as the attacking midfielders have 2.6 times chance of winning a match compared to all matches in general.

Whereas, the combination (in yellow) with Ramires in the attacking midfield position with Costa as the striker has more than 3 time chance of losing comapred to random chance.

We can therefore say that any three from the combination - Cesc Fabregas, Eden Hazard, Pedro Rodriguez or Willian - is a better fit when Diego costa is the Striker, and that Ramires should not be paired up with Costa.

2.3 Conclusion

From associations we can say that certain player combination might have an impact on the match result. ** The following will be the combination with highest chance of winning - **

Strikers Diego Costa

Mid field Eden Hazard, Willian, Pedro Rodríguez , Nemanja Matic, Cesc Fàbregas

Defenders John Terry , Branislav Ivanovic,Kurt Zouma, César Azpilicueta

GoalKeeper Thibaut Courtois

3. Does player positions impact match result?

Rationale

Analyzing groups like midfielders and defenders might not provide a complete picture without taking into account the positions in which these players play, since we see that players play in multiple positions over time. We hence analyzed midfielders and defenders separately for player positioning to result.

3.1 Midfielder positiong

Analysis

We got the cordinate information for each player-position of midfielders and used the apripori algorithm to look for rules that result in a win or loss

```
## Top teams
## home matches
match1<-collect(match_tbl)
matches_home <- subset(match1, (match1$home_team_api_id == 8455) & ((
  match1$season  == '2015/2016')))
matches_home$result<-
  case_when(matches_home$home_team_goal>matches_home$away_team_goal~"won",
matches_home$home_team_goal<matches_home$away_team_goal~"lost",
matches_home$home_team_goal==matches_home$away_team_goal~"draw" )
match_player<-select(matches_home,match_api_id ,home_team_api_id,away_team_api_id,
                     num_range("home_player_", 1:11),num_range("home_player_X", 1:11),
                     num_range("home_player_Y", 1:11), result)

## away matches
match2<-collect(match_tbl)
matches_away <- subset(match2, (match2$away_team_api_id == 8455) & ((
  match2$season  == '2015/2016')))
matches_away$result<-
  case_when(matches_away$home_team_goal<matches_away$away_team_goal~"won",
matches_away$home_team_goal>matches_away$away_team_goal~"lost",
matches_away$home_team_goal==matches_away$away_team_goal~"draw" )
match_player_away<-select(matches_away,match_api_id ,home_team_api_id,
away_team_api_id,num_range("away_player_", 1:11),num_range(
"away_player_X", 1:11),num_range("away_player_Y", 1:11), result)

## home games
playerName<-match_player %>%
  as.data.frame %>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,num_range(
    "home_player_", 1:11))%>%
  melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,variable,value)

playerX<-match_player %>%
  as.data.frame %>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,num_range(
    "home_player_X", 1:11))%>%
  melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
  select(variable, value)
```

```r
playerY<-match_player %>%
  as.data.frame %>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,num_range(
    "home_player_Y", 1:11))%>%
  melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
  select(variable, value)

player_home<-cbind(playerName,playerX,playerY)
names(player_home)<-c("match_api_id","Chelsea","against","result","player",
                      "player_id","playerx","player_xposn","playery","player_yposn")
player_home<-select(player_home, match_api_id, result,against,player_id,player_xposn,
                    player_yposn)
player_home$type<-"Home"


## getting player names

player_home<-merge(player_home,player,by.x = "player_id", by.y = "player_api_id")
player_home<-merge(player_home,team,by.x = "against", by.y = "team_api_id")
home_games<- select(player_home,match_api_id,result,team_long_name,player_name,player_id,
player_xposn,player_yposn)
home_games$playerposn<-paste(home_games$player_name, home_games$player_xposn,
home_games$player_yposn,
sep = "-")
home_games$val<-1


## Away games

## getting position for each player for home games
playerNameAway<-match_player_away %>%
  as.data.frame %>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,num_range("away_player_",
  1:11))%>%
  melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
  select(match_api_id, home_team_api_id, away_team_api_id,result,variable,value)

playerXaway<-match_player_away %>%
as.data.frame %>%
select(match_api_id, home_team_api_id, away_team_api_id,result,num_range("away_player_X",
1:11))%>%
melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
select(variable, value)

playerYaway<-match_player_away %>%
as.data.frame %>%
select(match_api_id, home_team_api_id, away_team_api_id,result,num_range("away_player_Y",
1:11))%>%
melt(., id = c('match_api_id', "home_team_api_id","away_team_api_id", "result"))%>%
select(variable, value)

player_away<-cbind(playerNameAway,playerXaway,playerYaway)
names(player_away)<-c("match_api_id","against","chelsea","result","player","player_id",
```

```r
"playerx",
"player_xposn","playery","player_yposn")
player_away<-select(player_away, match_api_id, result,against,player_id,player_xposn,
player_yposn)
player_away$type<-"Away"


## getting player names

player_away<-merge(player_away,player,by.x = "player_id", by.y = "player_api_id")
player_away<-merge(player_away,team,by.x = "against", by.y = "team_api_id")
away_games<- select(player_away, match_api_id,result,team_long_name,player_name,player_id,
                    player_xposn,player_yposn)
away_games$playerposn<-paste(away_games$player_name, away_games$player_xposn,
                             away_games$player_yposn, sep = "-")
away_games$val<-1


## combining home and away

home_away_games<-rbind(home_games, away_games)
write.csv(home_away_games,"Home_away_games_14-15.csv")

## filtering for midfielders

midfielders<-c(115067,  155066, 107417, 30613,  128864, 178812, 94086,  150250,
               604982,  32345,  467354, 79574)
home_away_games_mid_fielders<-subset(home_away_games,player_id %in% midfielders )
```

```r
## table for AR rules for player positiion

### creating table for arules

home_games_wide1<-home_away_games_mid_fielders%>%
  select(match_api_id, playerposn,val)%>%
  spread(playerposn,val, fill = 0)

home_games_wide2<-home_away_games_mid_fielders%>%
  select(match_api_id, result,val)%>%
  distinct%>%
  spread(result,val, fill = 0)


ap_in<-merge(home_games_wide1,home_games_wide2, by = "match_api_id")
trans<- as(as.matrix(ap_in[, -1]), 'transactions')

rules <- apriori(trans,parameter=list(support = 0.1,conf=0.1))
```

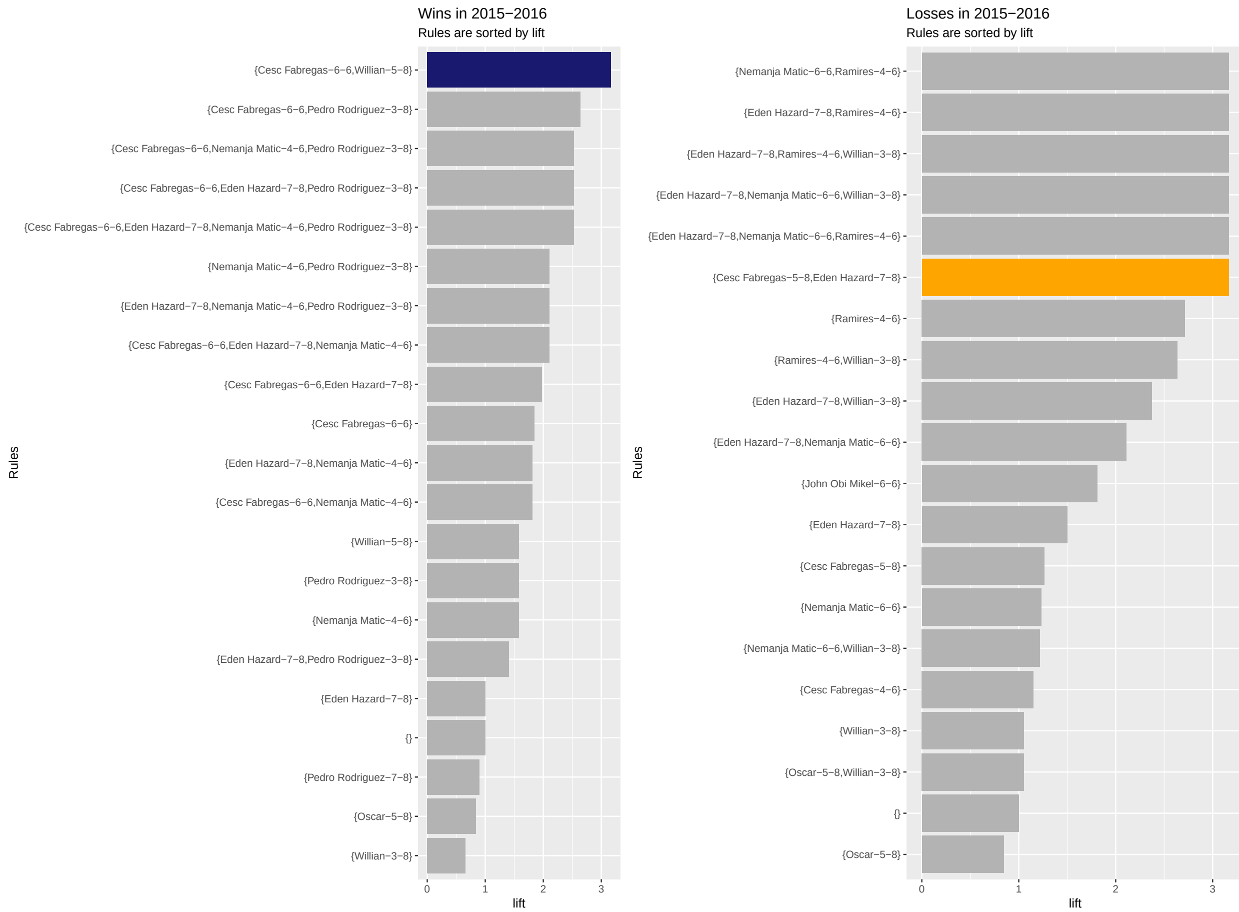
```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target   ext
```

```
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 3
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[27 item(s), 38 transaction(s)] done [0.00s].
## sorting and recoding items ... [17 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 done [0.00s].
## writing ... [365 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Comparing wins to losses

```
#Winning
win_rules<-subset(rules, subset = rhs %pin% "won" )

rule_win_df = data.frame(
       lhs = labels(lhs(win_rules)),
       rhs = labels(rhs(win_rules)),
       win_rules@quality)
rule_win_df<-rule_win_df[order(-rule_win_df$lift),]

rule_win_df$colors<-ifelse(rule_win_df$lhs ==
                             '{Cesc Fabregas-6-6,Willian-5-8}' | rule_win_df$lhs == '
                           {Cesc Fabregas-6-6,Eden Hazard-7-8,Nemanja Matic-4-6,
                           Pedro Rodriguez-3-8}' , "one" ,"two")
loss_rules<-subset(rules, subset = rhs %pin% "lost" )
rule_loss_df = data.frame(
       lhs = labels(lhs(loss_rules)),
       rhs = labels(rhs(loss_rules)),
       loss_rules@quality)
rule_loss_df<-rule_loss_df[order(-rule_loss_df$lift),]

rule_loss_df$colors<-ifelse(rule_loss_df$lhs == '{Eden Hazard-7-8,
Nemanja Matic-6-6,Ramires-4-6}'
|rule_loss_df$lhs == '{Cesc Fabregas-5-8,Eden Hazard-7-8}',"one", "two")

p1<-ggplot( rule_win_df , aes( x = reorder(lhs,lift) , y = lift,
fill = colors, label = lift )) +
geom_bar(stat="identity") + coord_flip() +scale_fill_manual(values = c(
"midnightblue","gray70")) +
theme(legend.position="none") + labs(title = "Wins in 2015-2016", y= "lift",
x = "Rules",
subtitle = "Rules are sorted by lift")

p2<-ggplot(rule_loss_df , aes( x = reorder(lhs,lift) , y = lift,fill = colors,
label = lift)) +
geom_bar(stat="identity") + coord_flip()+scale_fill_manual(values = c("orange",
"gray70")) +
theme(legend.position="none")+labs(title = "Losses in 2015-2016",y= "lift",
```

```
x ="Rules", subtitle = "Rules are sorted by lift")
grid.arrange(p1, p2,  ncol = 2, nrow = 1)
```

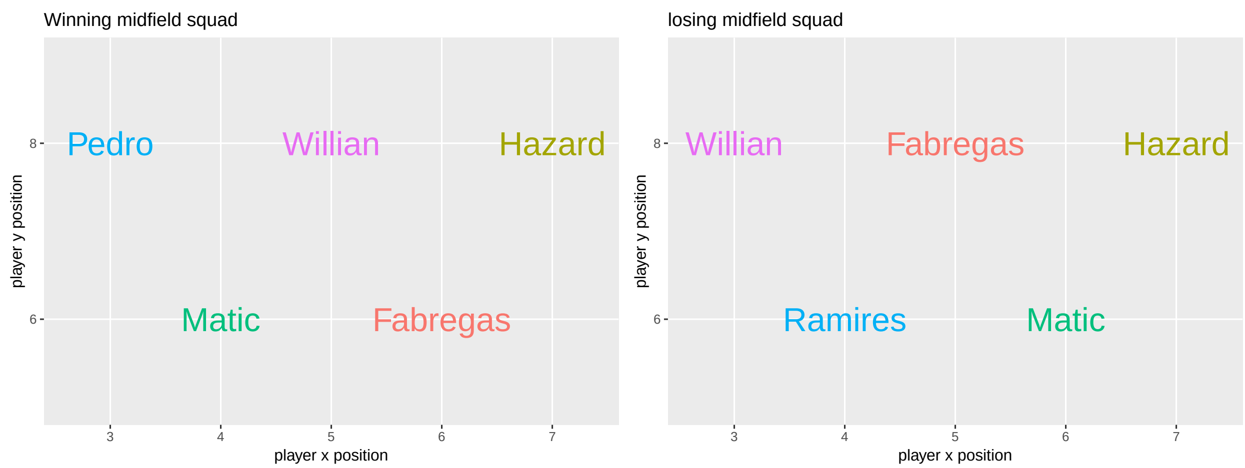


Interpretation

- For Cesc Fabregas (6,6) is a winning position with a win lift of 1.85. On contrary, him playing at (4,6) has a loss lift of 1.25.
- At central mid field position ((4,6),(6,6)) Fabregas and Mattic is the best possible combination.
- Willian playing (5,8) has higher chance of winning than Willian playing (3,8)
- Combination of Pedro Rodrigues (3,8), Eden Hazard (7,8), Nemanja Matic (4,6), Cesc Fabregas (6,6) had twice the chance of winning over Chelsea average win rate

```
library(ggplot2)
library(gridExtra)
x_win<-c(3,4,5,6,7)
y_win<-c(8,6,8,6,8)
p_win<-c("Pedro","Matic","Willian","Fabregas","Hazard")
p_loss<-c("Willian","Ramires","Fabregas","Matic","Hazard")

wins<-as.data.frame(cbind(x_win,y_win,p_win))
plt1<-ggplot(wins, aes(x = x_win, y = y_win, color = p_win, label = p_win)) +
geom_text(aes(label=p_win), size=8) + labs(title= "Winning midfield squad",
x = "player x position", y = "player y position") +
theme(legend.position="none")
```

```
loss<-as.data.frame(cbind(x_win,y_win,p_loss))
plt2<-ggplot(loss, aes(x = x_win, y = y_win, color = p_loss, label = p_loss)) +
geom_text(aes(label=p_loss), size=8) + labs(title= "losing midfield squad",
x = "player x position", y = "player y position") +
theme(legend.position="none")

grid.arrange(plt1, plt2,  ncol = 2)
```



Interpretation

The above figure is a representation of winning squad on the left and losing on the right. The colors for the players can be used to understand how their position changes.

3.2 Defender positioning

Analysis

We joined the player names to x and y position of defenders and used the apripori algorithm to look for rules that result in a win or loss

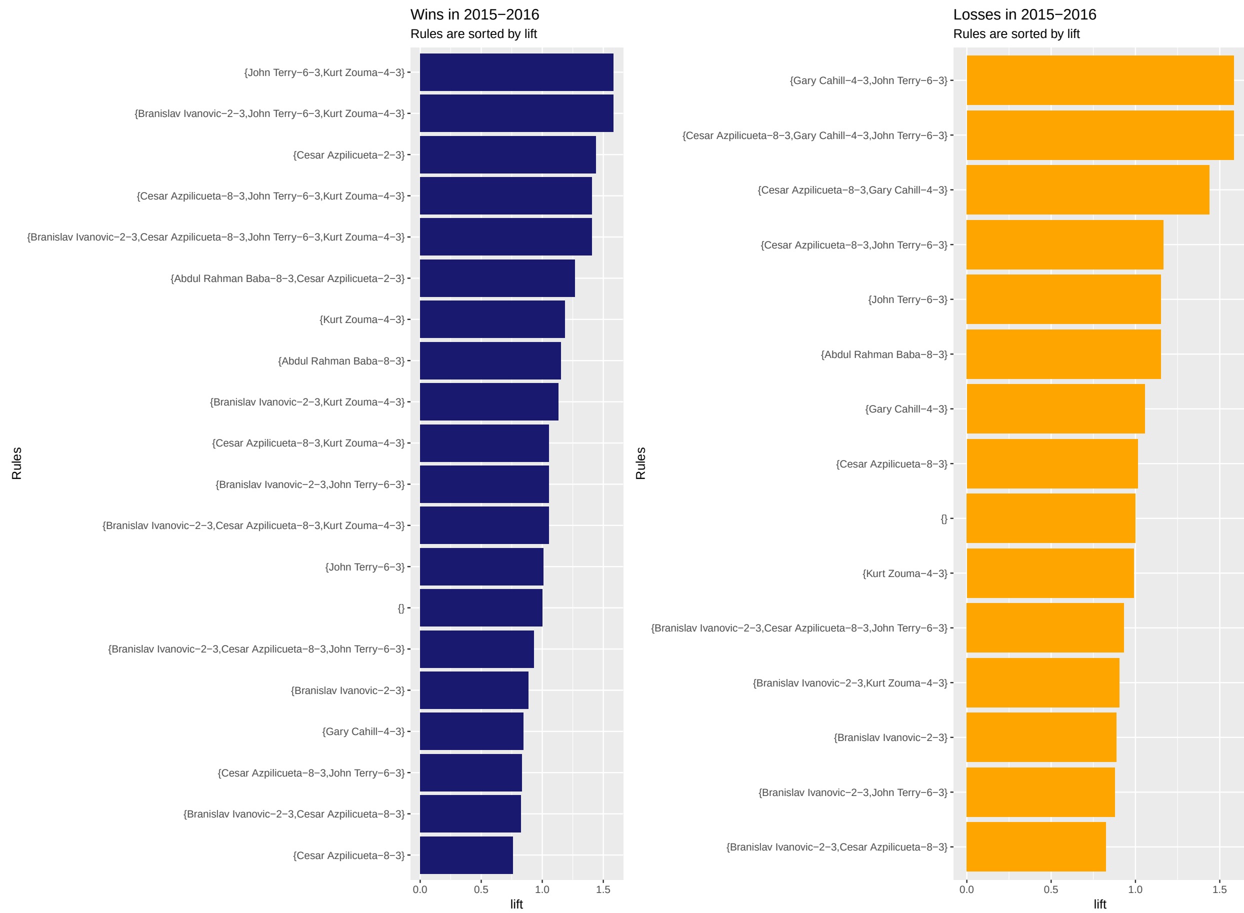
```
## filtering for defenders positions

defenders<-c(23783,281207,30627,31306,324910,72541, 41167)
home_away_games_defenders<-subset(home_away_games,player_id %in% defenders )

## table for AR rules for player positiion

### creating table for arules

home_games_wide1<-home_away_games_defenders%>%
  select(match_api_id, playerposn,val)%>%
  spread(playerposn,val, fill = 0)

home_games_wide2<-home_away_games_defenders%>%
  select(match_api_id, result,val)%>%
  distinct%>%
  spread(result,val, fill = 0)
```

```r
ap_in<-merge(home_games_wide1,home_games_wide2, by = "match_api_id")
trans<- as(as.matrix(ap_in[, -1]), 'transactions')

rules <- apriori(trans,parameter=list(support = 0.1,conf=0.1))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      1
##  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: 3
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[16 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 done [0.00s].
## writing ... [233 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
#Winning
win_rules<-subset(rules, subset = rhs %pin% "won" )

rule_win_df = data.frame(
       lhs = labels(lhs(win_rules)),
       rhs = labels(rhs(win_rules)),
       win_rules@quality)
rule_win_df<-rule_win_df[order(-rule_win_df$lift),]

rule_win_df$colors<-ifelse(rule_win_df$lhs == '{Cesc Fabregas-6-6,Willian-5-8}' |
                             rule_win_df$lhs == '{Cesc Fabregas-6-6,Eden Hazard-7-8,
                           Nemanja Matic-4-6,Pedro Rodriguez-3-8}' , "one" ,"two")
loss_rules<-subset(rules, subset = rhs %pin% "lost" )
rule_loss_df = data.frame(
       lhs = labels(lhs(loss_rules)),
       rhs = labels(rhs(loss_rules)),
       loss_rules@quality)
rule_loss_df<-rule_loss_df[order(-rule_loss_df$lift),]

rule_loss_df$colors<-ifelse(rule_loss_df$lhs == '{Eden Hazard-7-8,
                            Nemanja Matic-6-6,Ramires-4-6}'|rule_loss_df$lhs == '{
                            Cesc Fabregas-5-8,Eden Hazard-7-8}',"one", "two")


p1<-ggplot( rule_win_df , aes( x = reorder(lhs,lift) , y = lift,
                               fill = colors, label = lift ))+geom_bar(stat="identity")+
  coord_flip() +scale_fill_manual(values = c("midnightblue","gray70")) +
```

```r
  theme(legend.position="none") + labs(title = "Wins in 2015-2016", y= "lift", x = "Rules",
                                       subtitle = "Rules are sorted by lift")


p2<-ggplot( rule_loss_df,aes(x=reorder(lhs,lift),y=lift,fill=colors,label=lift))+
  geom_bar(stat="identity")+coord_flip()+scale_fill_manual(values=c("orange","gray70"))+
  theme(legend.position="none")+labs(title ="Losses in 2015-2016",y="lift",x = "Rules",
                                     subtitle = "Rules are sorted by lift")

grid.arrange(p1, p2,  ncol = 2, nrow = 1)
```



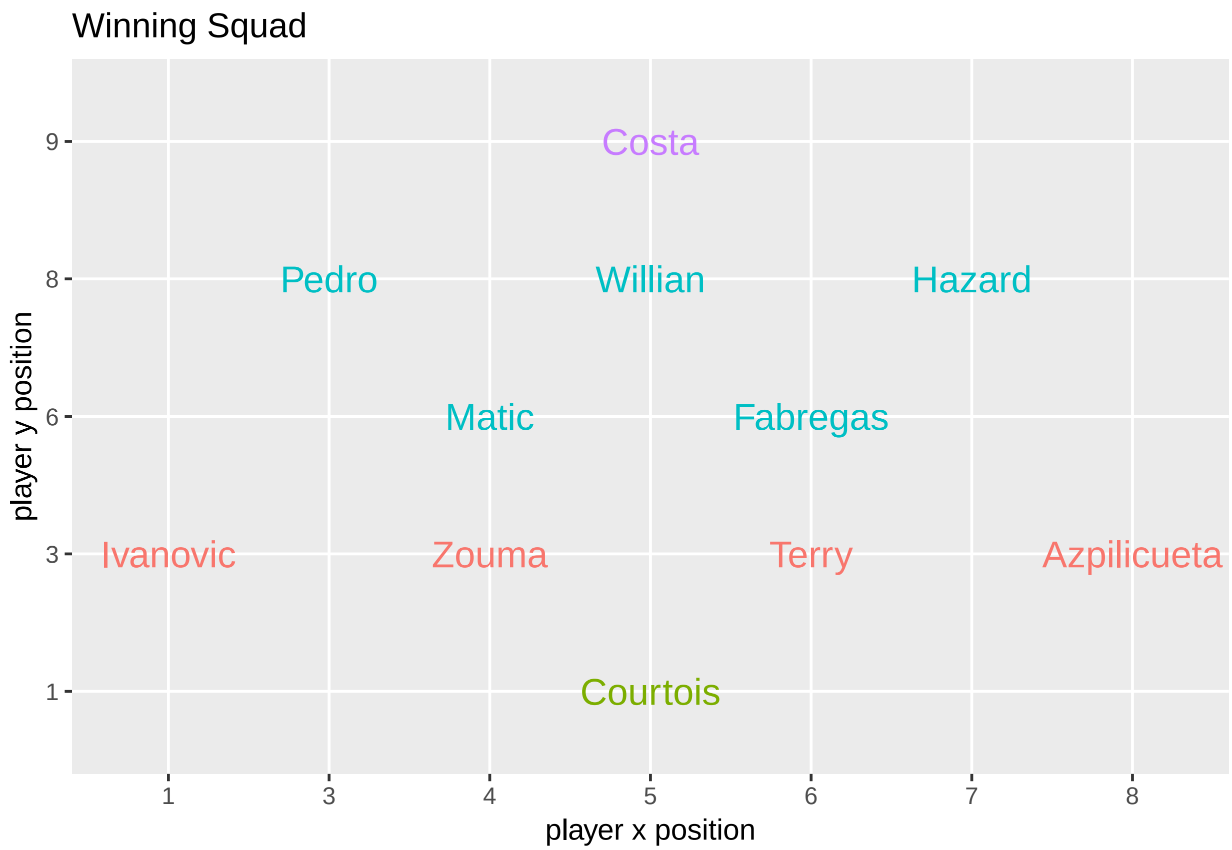
Interpretation

- We see that the center back pairing(cordinates:(6,3),(4,3)) of **Zouma (4,3) and Terry(6,3)** contributes to higher chance of winning over the combination of **Zouma (4,3) and Cahill(6,3)** .
- Even though Aspilicueta plays better at (2,3), the defense line seems to be stronger when Ivanovic is at (2,3) with Azpilicueta at (8,3), Zouma at (4,3) and Terry at (6,3) (in blue),

3.3 Final squad

```r
x_win<-c(3,4,5,6,7,5,5,1,4,6,8)
y_win<-c(8,6,8,6,8,9,1,3,3,3,3)
p_win<-c("Pedro","Matic","Willian","Fabregas","Hazard","Costa","Courtois","Ivanovic",
         "Zouma","Terry","Azpilicueta")
```

```r
position <- c('M','M','M','M','M','S','G','D','D','D','D')
wins<-as.data.frame(cbind(x_win,y_win,p_win))
plt_1<-ggplot(wins, aes(x = x_win, y = y_win,color =position, label = p_win)) +
  geom_text(aes(label=p_win),size=5)+labs(title= "Winning Squad",x="player x position",
                                          y = "player y position") +
  theme(legend.position="none")
plt_1
```




```r
#Pedro Rodrigues (3,8),Eden Hazard (7,8),Nemanja Matic (4,6), Cesc Fàbregas (6,6)
```

D. Cluster Analysis

While our analyses above gave insights on which players perform better from the existing squad, we will have to rethink our strategy in case some of the existing players move out or a new player comes in. Segmenting existing players into groups will provide direction on which player to purchase and also, where the new player might play.

To accomplish this, we had created a clustering algorithm which takes the player's game attributes, his age and his playing position (defender/midfielder/striker). Based on the clusters, we can identify which new player might replace an existing player, if the new player had not played for Chelsea before.

As the first step we are filtering the Match attributes table for Chelsea for home and away matches.

```r
long_team_name <- 'Chelsea'

# Filtering for Chelsea in the team table
myteam_team_tbl <- team_tbl %>% filter(grepl(long_team_name, team_long_name))
```

```
home_matches <- match_tbl %>% filter(home_team_api_id == myteam_team_tbl$team_api_id) %>%
  mutate(result = ifelse(home_team_goal > away_team_goal, "Win", ifelse(home_team_goal <
away_team_goal, "Loss", "Draw")))
away_matches <- match_tbl %>% filter(away_team_api_id == myteam_team_tbl$team_api_id) %>%
  mutate(result = ifelse(home_team_goal > away_team_goal, "Loss", ifelse(home_team_goal <
away_team_goal, "Win", "Draw")))
all_matches <- rbind(home_matches, away_matches)
```

Next we identify players for each match.

```
# List of players for home matches
home_matches_players <- select(home_matches, id, date, season, result, matches(
  "home_player_[[:digit:]]")) %>%
                gather(player, player_api_id, -id, -date, -result, -season) %>%
                rename(match_id = id)

home_matches_players$loc <- "home"
names <- names(home_matches_players)

# List of players for away matches
away_matches_players <- select(away_matches, id, date, season, result, matches(
  "away_player_[[:digit:]]")) %>% gather(player, player_api_id,-id,-date,-result,-season
                                         ) %>% rename(match_id = id)

away_matches_players$loc <- "away"

# List of all Chelsea players
all_matches_players <- rbind(home_matches_players, away_matches_players)

all_players <- all_matches_players %>% select(player_api_id) %>% distinct()
all_players <- merge(all_players, player_tbl[, c("player_api_id", "player_name")], by =
                       "player_api_id")
```

We obtain the player positions from the web for the list of players that played in Chelsea. Writing out the player names and reading the updated file with positions * G for Goalkeeper * S for Striker * M for Midfielder * D for Defender

```
# Reading Chelsea player positions file (S, D, M, G)
all_player_position <- read.csv('home_players_position.csv', stringsAsFactors = FALSE)
table(all_player_position$position)
```

```
##
##  D  G  M  S
## 23  7 27 15
```

Getting attributes of Chelsea players from the player attributes table

```
# Attriubtes for Chelsea players
all_player_attributes <- player_atts_tbl %>% filter(
  player_api_id %in% all_players$player_api_id)
```

We know whats here There are 3 missing values for all the scoring attributes like dribbling to gk_diving

Generating a matrixplot to identify where the missing values are present

Values are missing for the same record across all attributes. Filtering records for those players alone to understand the missing values

```r
null_id <- all_player_attributes[is.na(all_player_attributes$gk_diving), "player_api_id"]
all_player_attributes_null <- all_player_attributes %>% filter(player_api_id %in%
                                                          null_id$player_api_id) %>%
  arrange(player_api_id, date)
```

The missing values are from duplicates of an existing record. Since, we have attributes of players for the corresponding dates, we can delete the NA records

```r
all_player_attributes <- all_player_attributes %>% filter(!is.na(gk_diving))

# Updated summary of the player attributes to confirm there are no missing values
summary(all_player_attributes)
```

```
##        id          player_fifa_api_id player_api_id        date
##  Min.   :   858   Min.   :    27     Min.   : 19243   Length:1883
##  1st Qu.: 47250   1st Qu.: 49369     1st Qu.: 30679   Class :character
##  Median : 94740   Median :172723     Median : 41167   Mode  :character
##  Mean   : 96202   Mean   :137545     Mean   : 98037
##  3rd Qu.:144934   3rd Qu.:189505     3rd Qu.:150250
##  Max.   :182790   Max.   :215639     Max.   :604982
##  overall_rating    potential     preferred_foot     attacking_work_rate
##  Min.   :49.00   Min.   :59.00   Length:1883        Length:1883
##  1st Qu.:74.00   1st Qu.:79.00   Class :character   Class :character
##  Median :79.00   Median :82.00   Mode  :character   Mode  :character
##  Mean   :77.42   Mean   :81.73
##  3rd Qu.:82.00   3rd Qu.:85.00
##  Max.   :91.00   Max.   :93.00
##  defensive_work_rate    crossing       finishing     heading_accuracy
##  Length:1883         Min.   : 9.00   Min.   : 1.00   Min.   : 8.00
##  Class :character    1st Qu.:55.50   1st Qu.:46.00   1st Qu.:54.00
##  Mode  :character    Median :68.00   Median :65.00   Median :67.00
##                      Mean   :63.03   Mean   :59.73   Mean   :63.12
##                      3rd Qu.:77.00   3rd Qu.:78.00   3rd Qu.:75.00
##                      Max.   :91.00   Max.   :95.00   Max.   :95.00
##  short_passing      volleys        dribbling         curve
##  Min.   :23.00   Min.   : 9.00   Min.   : 6.00   Min.   : 8.00
##  1st Qu.:66.00   1st Qu.:47.00   1st Qu.:61.00   1st Qu.:56.00
##  Median :74.00   Median :66.00   Median :76.00   Median :68.00
##  Mean   :70.12   Mean   :59.58   Mean   :68.67   Mean   :62.19
##  3rd Qu.:79.00   3rd Qu.:77.00   3rd Qu.:83.00   3rd Qu.:77.00
##  Max.   :96.00   Max.   :93.00   Max.   :94.00   Max.   :92.00
##  free_kick_accuracy  long_passing    ball_control    acceleration
##  Min.   :10.00      Min.   :14.00   Min.   :16.00   Min.   :23.00
##  1st Qu.:49.00      1st Qu.:56.00   1st Qu.:66.00   1st Qu.:68.00
##  Median :59.00      Median :65.00   Median :78.00   Median :77.00
##  Mean   :56.73      Mean   :63.32   Mean   :72.25   Mean   :73.76
##  3rd Qu.:71.00      3rd Qu.:74.00   3rd Qu.:83.00   3rd Qu.:84.00
##  Max.   :92.00      Max.   :95.00   Max.   :94.00   Max.   :95.00
##   sprint_speed      agility        reactions         balance
##  Min.   :26.00   Min.   :36.0   Min.   :42.00   Min.   :34.00
##  1st Qu.:68.00   1st Qu.:63.0   1st Qu.:70.00   1st Qu.:59.50
##  Median :77.00   Median :74.0   Median :75.00   Median :71.00
##  Mean   :73.71   Mean   :70.8   Mean   :75.02   Mean   :67.71
##  3rd Qu.:83.00   3rd Qu.:81.0   3rd Qu.:81.00   3rd Qu.:78.00
```

```
##  Max.   :94.00   Max.   :92.0   Max.   :96.00   Max.   :90.00
##    shot_power       jumping         stamina         strength
##  Min.   :20.00   Min.   :38.00   Min.   :19.00   Min.   :27.00
##  1st Qu.:62.00   1st Qu.:63.00   1st Qu.:66.00   1st Qu.:64.00
##  Median :73.00   Median :70.00   Median :73.00   Median :72.00
##  Mean   :68.34   Mean   :70.07   Mean   :70.68   Mean   :70.52
##  3rd Qu.:80.00   3rd Qu.:77.00   3rd Qu.:78.00   3rd Qu.:79.00
##  Max.   :93.00   Max.   :93.00   Max.   :96.00   Max.   :96.00
##    long_shots     aggression    interceptions    positioning
##  Min.   : 7.0   Min.   :17.00   Min.   :15.00   Min.   : 8.00
##  1st Qu.:53.0   1st Qu.:52.00   1st Qu.:39.00   1st Qu.:52.00
##  Median :67.0   Median :68.00   Median :59.00   Median :70.00
##  Mean   :61.9   Mean   :63.87   Mean   :55.74   Mean   :64.62
##  3rd Qu.:76.0   3rd Qu.:76.00   3rd Qu.:74.00   3rd Qu.:81.00
##  Max.   :95.0   Max.   :93.00   Max.   :91.00   Max.   :95.00
##      vision         penalties         marking      standing_tackle
##  Min.   :14.00   Min.   : 8.00   Min.   : 6.00   Min.   : 7.00
##  1st Qu.:56.00   1st Qu.:53.00   1st Qu.:23.00   1st Qu.:25.00
##  Median :70.00   Median :67.00   Median :39.00   Median :48.00
##  Mean   :66.37   Mean   :63.79   Mean   :46.95   Mean   :50.63
##  3rd Qu.:78.00   3rd Qu.:77.00   3rd Qu.:72.00   3rd Qu.:75.00
##  Max.   :93.00   Max.   :94.00   Max.   :93.00   Max.   :95.00
##  sliding_tackle    gk_diving      gk_handling     gk_kicking
##  Min.   : 8.00   Min.   : 1.00   Min.   : 2.0   Min.   : 2.00
##  1st Qu.:24.00   1st Qu.: 8.00   1st Qu.: 8.0   1st Qu.: 8.00
##  Median :45.00   Median :10.00   Median :12.0   Median :12.00
##  Mean   :48.72   Mean   :15.31   Mean   :17.1   Mean   :22.99
##  3rd Qu.:73.50   3rd Qu.:13.00   3rd Qu.:15.0   3rd Qu.:15.00
##  Max.   :92.00   Max.   :94.00   Max.   :91.0   Max.   :95.00
##  gk_positioning   gk_reflexes
##  Min.   : 1.00   Min.   : 2.0
##  1st Qu.: 7.00   1st Qu.: 9.0
##  Median :12.00   Median :12.0
##  Mean   :16.85   Mean   :17.6
##  3rd Qu.:15.00   3rd Qu.:15.0
##  Max.   :91.00   Max.   :94.0
```

Adding season to the dates in which the players were assigned these attributes. This enables us to map the player's attributes with the matches corresponding to the season in which he played in.

```r
# Identifying season
all_player_attributes$season<-as.numeric(ceiling((
  lubridate::date(all_player_attributes$date)- as.Date('2008-07-01')) / 365))
all_player_attributes$season <- ifelse(all_player_attributes$season <= 0, 1,
                                       all_player_attributes$season)
all_player_attributes$season <- paste(all_player_attributes$season + 2007,
                                      all_player_attributes$season + 2008, sep = "/")

agg_player_attribute=all_player_attributes%>%select(-preferred_foot,-attacking_work_rate,
-defensive_work_rate, -id, -player_fifa_api_id, -date) %>% group_by(
  player_api_id, season) %>% summarise_all(mean)
```

Identifying list of players by season and mapping their position and physical attributes

```r
plyr_season_map <- all_matches_players[, c("player_api_id", "season")]
plyr_season_map <- plyr_season_map[!duplicated(plyr_season_map), ]
```

```r
# Getting player position attributes (G, S, D, M) and physical attributes
agg_player_attribute_sub <- merge(merge(merge(agg_player_attribute, plyr_season_map),
all_player_position[, c(1, 3)]), player_tbl[, c('player_api_id', 'birthday', 'height',
'weight')])

# QC
table(plyr_season_map$season)
```

```
##
## 2008/2009 2009/2010 2010/2011 2011/2012 2012/2013 2013/2014 2014/2015
##        24        23        22        27        23        25        21
## 2015/2016
##        24
```

```r
table(agg_player_attribute_sub$season)
```

```
##
## 2008/2009 2009/2010 2010/2011 2011/2012 2012/2013 2013/2014 2014/2015
##        24        23        22        27        23        25        21
## 2015/2016
##        24
```

```r
# Adding age column and ordering
agg_player_attribute_sub$age<-as.numeric(floor((as.Date('2008-07-01')-lubridate::date(
  agg_player_attribute_sub$birthday)) / 365)) + as.numeric(substr
(agg_player_attribute_sub$season, 1, 4)) - 2007
agg_player_attribute_sub$birthday <- NULL
agg_player_attribute_sub$id <- paste(agg_player_attribute_sub$player_api_id,
                                     agg_player_attribute_sub$season, sep = '-')
agg_player_attribute_sub <- agg_player_attribute_sub[order(
  agg_player_attribute_sub$player_api_id, agg_player_attribute_sub$season), ]
```

There are over 35 attributes for each player. Combining them into groups will simplify interpretation and the reduction of dimensions in the data. The average of the metrics for each group was taken as the simplified representation. The following groups were constructed with the corresponding attributes:

- Attacking - Crossing, Finishing, Heading Accuracy, Short Passing and Volleys
- Defensive - Marking, Standing Tackle and Sliding Tackle
- Goalkeeper - GK Diving, GK Kicking, GK Handling, GK Positioning and GK Reflexes
- Metality - Aggression, Interceptions, Positioning, Vision and Penalties
- Movement - Acceleration, Sprint Speed, Agility, Reactions and Balance
- Power - Shot Power, Jumping, Stamina, Strength and Long Shots
- Skill - Dribbling, Curve, Free Kick Accuracy, Long Passing and Ball Control

```r
# Feature engineering by creating derived metrics to caputure gameplay of the player
agg_player_attribute_sub<-agg_player_attribute_sub%>%mutate(attacking=(
  crossing + finishing + heading_accuracy + short_passing + volleys) / 5
  , movement = (acceleration + sprint_speed + agility + reactions + balance) / 5
  , skill = (dribbling +curve + free_kick_accuracy + long_passing + ball_control) / 5,
  defensive = (marking +standing_tackle + sliding_tackle) / 3, mentality = (aggression +
  interceptions + positioning + vision + penalties) / 5, goalkeeper = (gk_diving +
  gk_kicking + gk_handling + gk_positioning +gk_reflexes) / 5, power = (shot_power +
  jumping + stamina + strength + long_shots) / 5)

# Subsetting for required columns only
player_attributes <- agg_player_attribute_sub %>% select(player_api_id, season,
```

```
id, position,overall_rating, potential, height, weight, age, attacking, movement,
skill, defensive, mentality, goalkeeper, power)
summary(player_attributes)
```

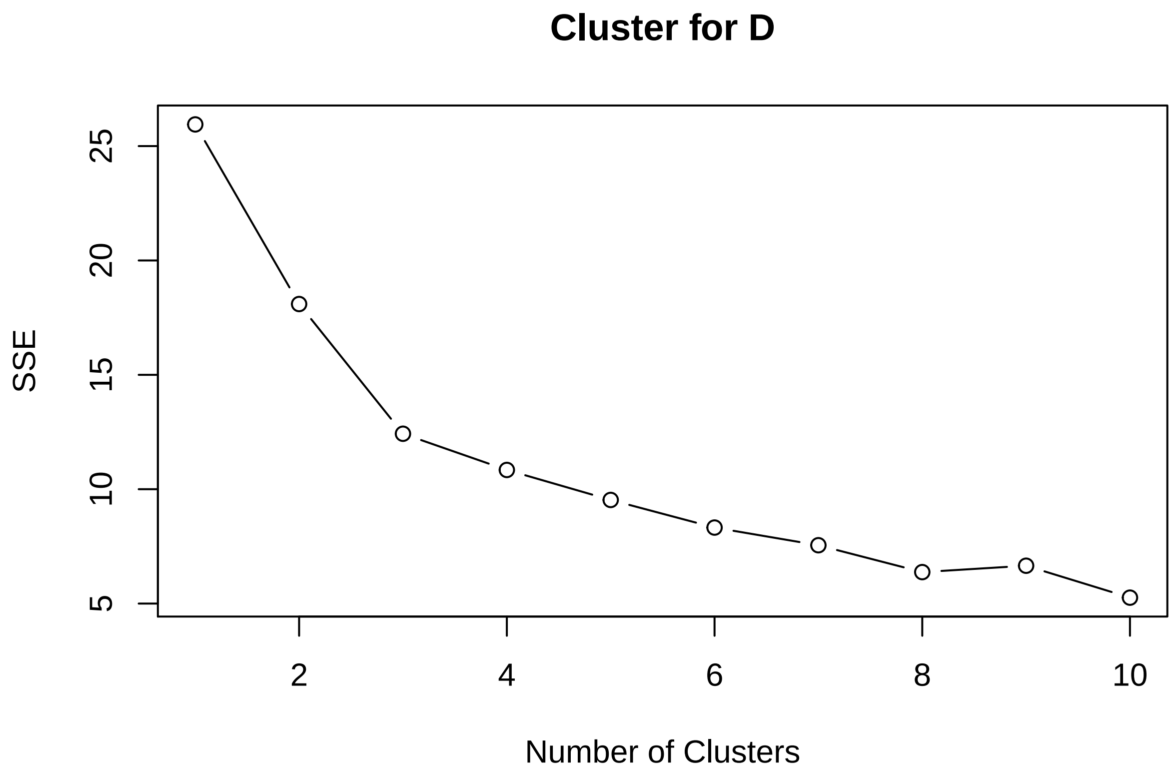
```
##  player_api_id       season               id              position
##  Min.   : 19243   Length:189         Length:189         Length:189
##  1st Qu.: 30675   Class :character   Class :character   Class :character
##  Median : 32345   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 72555
##  3rd Qu.: 72541
##  Max.   :604982
##  overall_rating    potential         height          weight
##  Min.   :56.67   Min.   :67.00   Min.   :167.6   Min.   :137.0
##  1st Qu.:79.00   1st Qu.:82.00   1st Qu.:177.8   1st Qu.:163.0
##  Median :81.50   Median :84.00   Median :182.9   Median :176.0
##  Mean   :80.54   Mean   :84.13   Mean   :183.6   Mean   :175.8
##  3rd Qu.:84.00   3rd Qu.:87.00   3rd Qu.:188.0   3rd Qu.:190.0
##  Max.   :88.75   Max.   :91.25   Max.   :200.7   Max.   :209.0
##       age          attacking        movement         skill
##  Min.   :18.00   Min.   :15.20   Min.   :42.40   Min.   :16.80
##  1st Qu.:24.00   1st Qu.:59.30   1st Qu.:66.20   1st Qu.:58.20
##  Median :28.00   Median :67.83   Median :74.90   Median :69.40
##  Mean   :27.43   Mean   :64.21   Mean   :72.73   Mean   :64.79
##  3rd Qu.:31.00   3rd Qu.:75.77   3rd Qu.:81.00   3rd Qu.:75.60
##  Max.   :41.00   Max.   :85.85   Max.   :89.60   Max.   :88.00
##    defensive       mentality       goalkeeper        power
##  Min.   :12.00   Min.   :22.00   Min.   : 6.80   Min.   :36.20
##  1st Qu.:26.13   1st Qu.:61.40   1st Qu.: 9.60   1st Qu.:66.20
##  Median :63.50   Median :68.00   Median :11.40   Median :73.00
##  Mean   :55.27   Mean   :66.36   Mean   :20.51   Mean   :70.28
##  3rd Qu.:80.08   3rd Qu.:75.50   3rd Qu.:25.80   3rd Qu.:77.20
##  Max.   :90.50   Max.   :87.60   Max.   :88.20   Max.   :87.20
```

Assuming that there are groups of players within each player type who exhibit similar behaviour within the group, and different behaviours across the groups, we can cluster players within the player types. Therefore, we can find clusters for strikers, defenders and midfielders. Since there are very few goalkeepers who had played for Chelsea across the years, we can focus on looking at only the strikers, defenders and midfielders. These clusters would be useful in understanding who might make a better replacement in case an existing player is leaving the squad or injured.

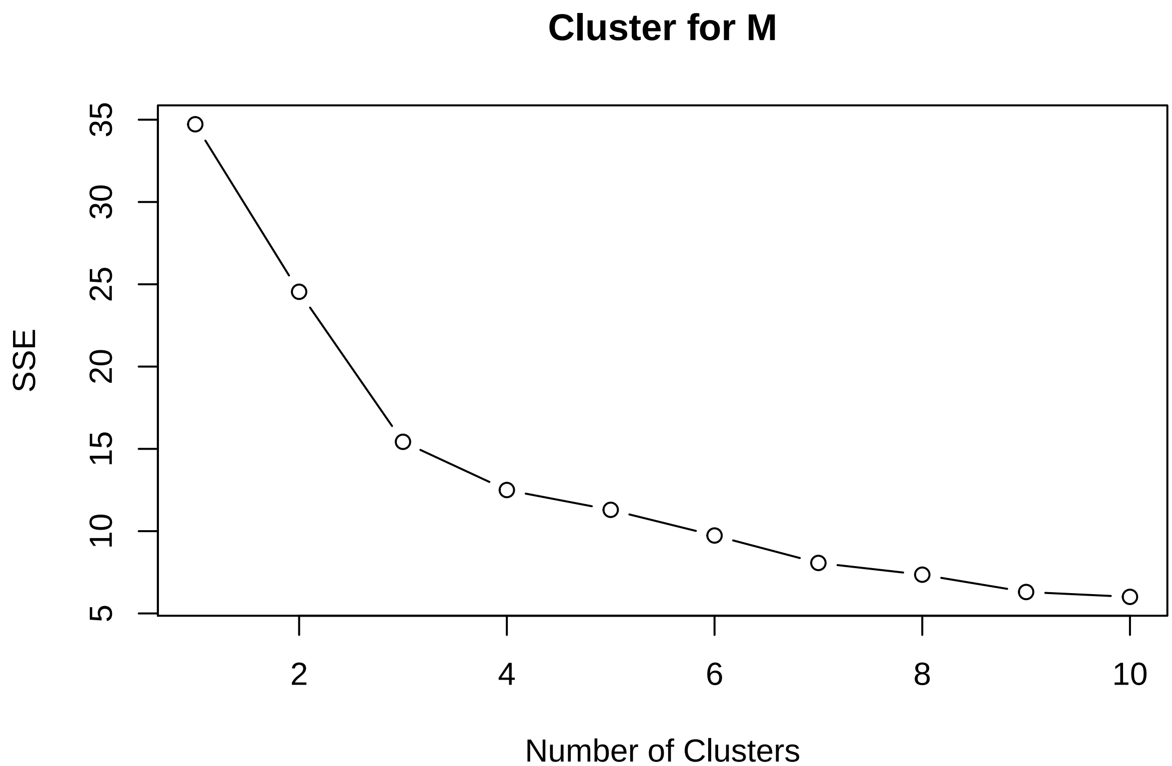
We have to normalize the data in order to cluster because there are certain attributes which are distriubted over different values.

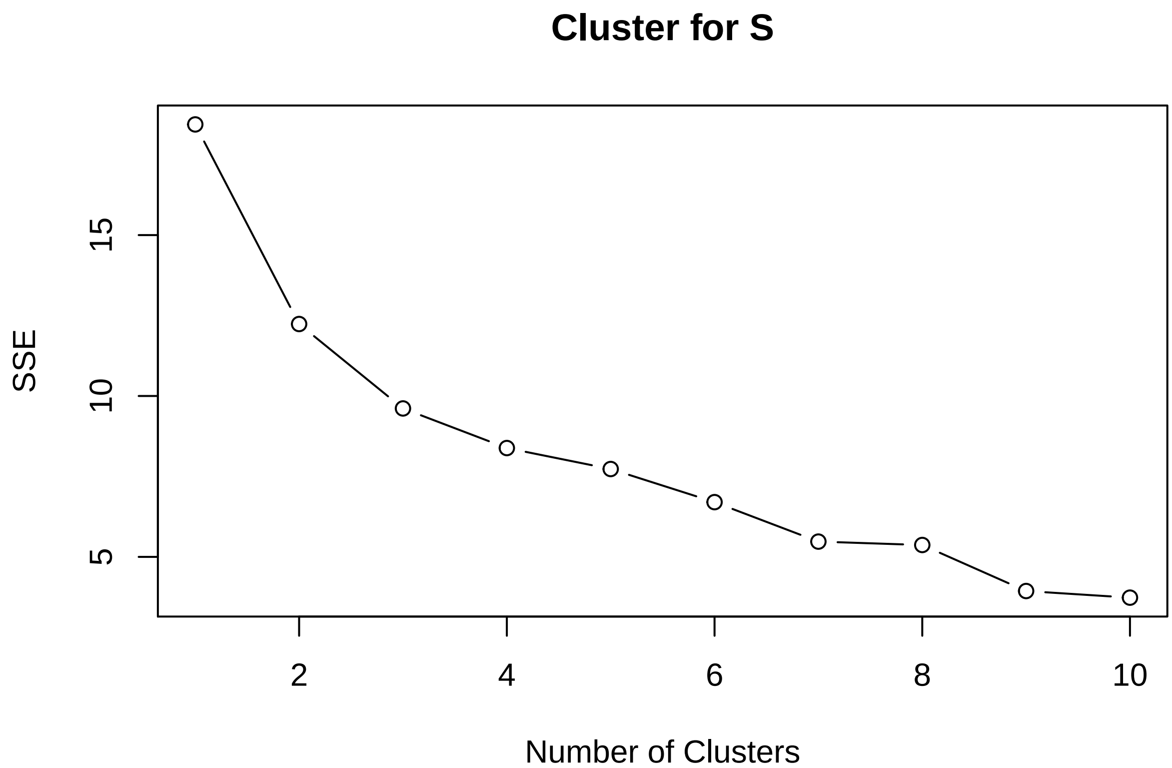
```
# Min-Max normalization function
normalize <- function(x){
  return ((x - min(x))/(max(x) - min(x)))}

# Identifying player types for sequential running
positions_for_loop <- c('D', 'M', 'S')

# Identifying the best k value from the elbow charts
for (i in 1:length(positions_for_loop))
{
  # Subsetting for the player type
```

```
  player_subset <- player_attributes %>% filter(position == positions_for_loop[i])

  # Normalizing the data
  data_normalized = mutate(player_subset,
                          overall_rating = normalize(overall_rating),
                          potential = normalize(potential),
                          height = normalize(height),
                          weight = normalize(weight),
                          age = normalize(age),
                          attacking = normalize(attacking),
                          movement = normalize(movement),
                          skill = normalize(skill),
                          defensive = normalize(defensive),
                          mentality = normalize(mentality),
                          goalkeeper = normalize(goalkeeper),
                          power = normalize(power))

  # Selecting the required columns for clustering
  norm_for_clust <- data_normalized %>% select(-player_api_id, -position, -season, -id
                                              , -height,-weight, -goalkeeper)
  rownames(norm_for_clust) <- data_normalized$id

  # k-means clustering to find the best k
  SSE_curve <- c()
  for (n in 1:10) {
    kcluster <- kmeans(norm_for_clust, n)
    #print(kcluster$withinss)
    sse <- sum(kcluster$withinss)
    SSE_curve[n] <- sse
  }
  # Plotting the SSE Curve for identifying k
  eval(parse(text = paste('plot(1:10, SSE_curve, type="b", xlab="Number of Clusters",
ylab="SSE",
                         main = "Cluster for ', '")',sep     =positions_for_loop[i])))
}
```



Cluster for D
SSE
25
20
15
10
5
2
4
6
8
10
Number of Clusters


Cluster for M
SSE
35
30
25
20
15
10
5
2
4
6
8
10
Number of Clusters



Based on the elbow curve, the following k values have been chosen for each player type:

- Strikers - 2 clusters
- Defenders - 3 clusters
- Midfielders - 3 clusters

```
# Obtaining the clusters for the different positions
cluster_k_values <- c(3, 3, 2)
player_cluster <- NULL

# Creating the clusters using the k-values obtained above
for (i in 1:length(positions_for_loop))
{
  player_subset <- player_attributes %>% filter(position == positions_for_loop[i])
  data_normalized = mutate(player_subset,
                           overall_rating = normalize(overall_rating),
                           potential = normalize(potential),
                           height = normalize(height),
                           weight = normalize(weight),
                           age = normalize(age),
                           attacking = normalize(attacking),
                           movement = normalize(movement),
                           skill = normalize(skill),
                           defensive = normalize(defensive),
                           mentality = normalize(mentality),
                           goalkeeper = normalize(goalkeeper),
                           power = normalize(power))
```

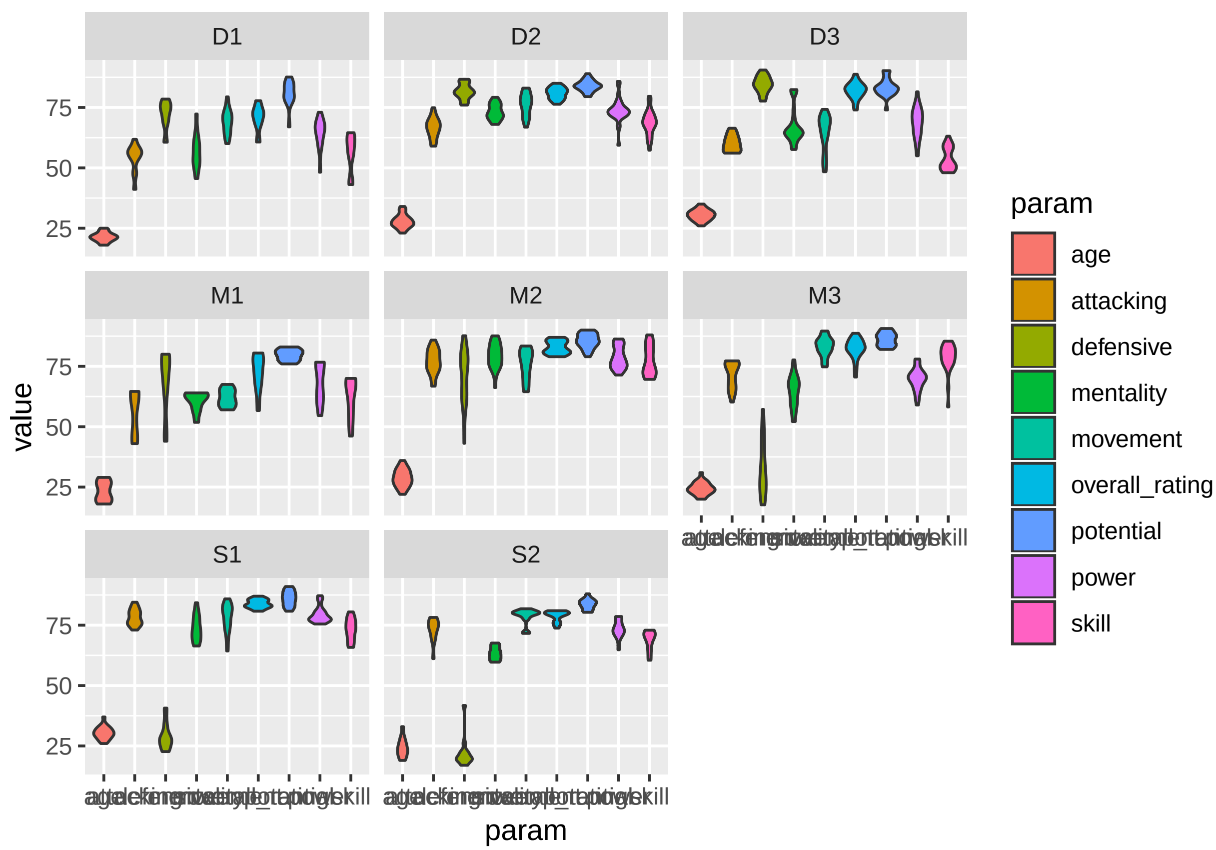
```
  norm_for_clust <- data_normalized %>% select(-player_api_id, -position, -season, -id,
                                        -height,          -weight, -goalkeeper)
  rownames(norm_for_clust) <- data_normalized$id

  # k-means clustering
  kcluster <- kmeans(norm_for_clust, cluster_k_values[i])
  kcluster_output <- data.frame(id = names(kcluster$cluster),
  cluster = paste(positions_for_loop[i],kcluster$cluster, sep = ""), row.names = NULL)

  player_cluster <- rbind(player_cluster, kcluster_output)
}
```

Joining the cluster with the player attributes to profile the clusters

```
agg_player_cluster <- merge(player_attributes, player_cluster, by="id", all.y = TRUE) %>%
  select(-player_api_id, -position, -season, -id, -height, -weight, -goalkeeper)
table(agg_player_cluster$cluster)
```

```
##
## D1 D2 D3 M1 M2 M3 S1 S2
## 13 30 24 10 31 27 22 14
```

```
# Plotting the distribution of metrics for each cluster to understand profiles
plot_data <- tidyr::gather(agg_player_cluster,param,value,-cluster)
ggplot(data = plot_data,aes(x=param,y=value,group=param,fill=param)) + geom_violin() +
  facet_wrap(~cluster)
```



Profiles

- Defenders:
  - D1 - age >= 25, movement > 70 and skill > 67.5 - older players with good skill and movement
  - D2 - age < 25 - Typically low on skill and mentality compared to other groups
  - D3 - age >= 25, movement < 70 and skill < 67.5 - older players with lower movement and lesser skill
- Midfielders:
  - M1 - age < 23 - Typically low on skill and mentality compared to other groups
  - M2 - age > 23, defensive > 50 and power > 73 - older players who perform well in terms of their defensive attributes and have greater power attributes
  - M3 - age > 23, defensive < 50 and power < 73 - older players who perform well in assisting the strikers. They have lower defending attributes, but assist strikers in terms of their attacking capabilities
- Defenders:
  - S1 - attacking > 75, mentality > 65 - Strong and experienced strikers able to adapt to the game and perform well
  - S2 - attacking < 75, mentality < 65 - Good strikers, but have a lower average age, indicating lower experience, but have potential on par with the experienced strikers

E. Appendix

```r
summary(match)
```

```
##        id          country_id      league_id        season
##  Min.   :    1   Min.   :    1   Min.   :    1   Length:25979
##  1st Qu.: 6496   1st Qu.: 4769   1st Qu.: 4769   Class :character
##  Median :12990   Median :10257   Median :10257   Mode  :character
##  Mean   :12990   Mean   :11739   Mean   :11739
##  3rd Qu.:19485   3rd Qu.:17642   3rd Qu.:17642
##  Max.   :25979   Max.   :24558   Max.   :24558
##
##      stage           date            match_api_id     home_team_api_id
##  Min.   : 1.00   Length:25979       Min.   : 483129   Min.   :  1601
##  1st Qu.: 9.00   Class :character   1st Qu.: 768437   1st Qu.:  8475
##  Median :18.00   Mode  :character   Median :1147511   Median :  8697
##  Mean   :18.24                      Mean   :1195429   Mean   :  9984
##  3rd Qu.:27.00                      3rd Qu.:1709853   3rd Qu.:  9925
##  Max.   :38.00                      Max.   :2216672   Max.   :274581
##
##  away_team_api_id home_team_goal   away_team_goal  home_player_X1
##  Min.   :  1601   Min.   : 0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:  8475   1st Qu.: 1.000   1st Qu.:0.000   1st Qu.:1.0000
##  Median :  8697   Median : 1.000   Median :1.000   Median :1.0000
##  Mean   :  9984   Mean   : 1.545   Mean   :1.161   Mean   :0.9996
##  3rd Qu.:  9925   3rd Qu.: 2.000   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :274581   Max.   :10.000   Max.   :9.000   Max.   :2.0000
##                                                    NA's   :1821
##  home_player_X2  home_player_X3  home_player_X4  home_player_X5
##  Min.   :0.000   Min.   :1.000   Min.   :2.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:4.000   1st Qu.:6.000   1st Qu.:8.000
```

```
##  Median :2.000   Median :4.000   Median :6.000   Median :8.000
##  Mean   :2.074   Mean   :4.061   Mean   :6.049   Mean   :7.545
##  3rd Qu.:2.000   3rd Qu.:4.000   3rd Qu.:6.000   3rd Qu.:8.000
##  Max.   :8.000   Max.   :8.000   Max.   :8.000   Max.   :9.000
##  NA's   :1821    NA's   :1832    NA's   :1832    NA's   :1832
##  home_player_X6  home_player_X7 home_player_X8 home_player_X9
##  Min.   :1.000   Min.   :1.00   Min.   :1.00   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:4.00   1st Qu.:3.00   1st Qu.:5.000
##  Median :3.000   Median :5.00   Median :6.00   Median :5.000
##  Mean   :3.185   Mean   :4.77   Mean   :5.31   Mean   :5.822
##  3rd Qu.:4.000   3rd Qu.:6.00   3rd Qu.:7.00   3rd Qu.:8.000
##  Max.   :9.000   Max.   :9.00   Max.   :9.00   Max.   :9.000
##  NA's   :1832    NA's   :1832   NA's   :1832   NA's   :1832
##  home_player_X10 home_player_X11 away_player_X1 away_player_X2
##  Min.   :1.000   Min.   :1.000   Min.   :1      Min.   :1.000
##  1st Qu.:4.000   1st Qu.:5.000   1st Qu.:1      1st Qu.:2.000
##  Median :5.000   Median :6.000   Median :1      Median :2.000
##  Mean   :5.389   Mean   :5.783   Mean   :1      Mean   :2.075
##  3rd Qu.:7.000   3rd Qu.:6.000   3rd Qu.:1      3rd Qu.:2.000
##  Max.   :9.000   Max.   :7.000   Max.   :6      Max.   :8.000
##  NA's   :1832    NA's   :1832    NA's   :1832   NA's   :1832
##  away_player_X3  away_player_X4  away_player_X5  away_player_X6
##  Min.   :2.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:4.000   1st Qu.:6.000   1st Qu.:8.000   1st Qu.:2.000
##  Median :4.000   Median :6.000   Median :8.000   Median :3.000
##  Mean   :4.059   Mean   :6.052   Mean   :7.526   Mean   :3.195
##  3rd Qu.:4.000   3rd Qu.:6.000   3rd Qu.:8.000   3rd Qu.:4.000
##  Max.   :9.000   Max.   :8.000   Max.   :9.000   Max.   :9.000
##  NA's   :1832    NA's   :1832    NA's   :1832    NA's   :1832
##  away_player_X7  away_player_X8  away_player_X9  away_player_X10
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:4.000   1st Qu.:3.000   1st Qu.:5.000   1st Qu.:4.000
##  Median :5.000   Median :6.000   Median :5.000   Median :5.000
##  Mean   :4.743   Mean   :5.294   Mean   :5.808   Mean   :5.476
##  3rd Qu.:6.000   3rd Qu.:7.000   3rd Qu.:8.000   3rd Qu.:7.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##  NA's   :1832    NA's   :1832    NA's   :1833    NA's   :1833
##  away_player_X11 home_player_Y1   home_player_Y2  home_player_Y3
##  Min.   :3.000   Min.   :0.0000   Min.   :0.000   Min.   :3
##  1st Qu.:5.000   1st Qu.:1.0000   1st Qu.:3.000   1st Qu.:3
##  Median :6.000   Median :1.0000   Median :3.000   Median :3
##  Mean   :5.766   Mean   :0.9996   Mean   :2.999   Mean   :3
##  3rd Qu.:6.000   3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:3
##  Max.   :8.000   Max.   :3.0000   Max.   :3.000   Max.   :5
##  NA's   :1839    NA's   :1821     NA's   :1821    NA's   :1832
##  home_player_Y4 home_player_Y5  home_player_Y6  home_player_Y7
##  Min.   :3      Min.   :3.000   Min.   :3.000   Min.   :3.000
##  1st Qu.:3      1st Qu.:3.000   1st Qu.:6.000   1st Qu.:6.000
##  Median :3      Median :3.000   Median :7.000   Median :7.000
##  Mean   :3      Mean   :3.237   Mean   :6.477   Mean   :6.672
##  3rd Qu.:3      3rd Qu.:3.000   3rd Qu.:7.000   3rd Qu.:7.000
##  Max.   :5      Max.   :8.000   Max.   :9.000   Max.   :9.000
##  NA's   :1832   NA's   :1832    NA's   :1832    NA's   :1832
##  home_player_Y8   home_player_Y9   home_player_Y10  home_player_Y11
```

```
##  Min.   : 3.000   Min.   : 1.000   Min.   : 3.000   Min.   : 1.00
##  1st Qu.: 7.000   1st Qu.: 7.000   1st Qu.: 8.000   1st Qu.:10.00
##  Median : 7.000   Median : 8.000   Median :10.000   Median :10.00
##  Mean   : 7.239   Mean   : 8.026   Mean   : 9.219   Mean   :10.44
##  3rd Qu.: 8.000   3rd Qu.: 8.000   3rd Qu.:10.000   3rd Qu.:11.00
##  Max.   :10.000   Max.   :10.000   Max.   :11.000   Max.   :11.00
##  NA's   :1832     NA's   :1832     NA's   :1832     NA's   :1832
##  away_player_Y1 away_player_Y2 away_player_Y3 away_player_Y4
##  Min.   :1      Min.   :3      Min.   :3      Min.   :3
##  1st Qu.:1      1st Qu.:3      1st Qu.:3      1st Qu.:3
##  Median :1      Median :3      Median :3      Median :3
##  Mean   :1      Mean   :3      Mean   :3      Mean   :3
##  3rd Qu.:1      3rd Qu.:3      3rd Qu.:3      3rd Qu.:3
##  Max.   :3      Max.   :3      Max.   :7      Max.   :7
##  NA's   :1832   NA's   :1832   NA's   :1832   NA's   :1832
##  away_player_Y5  away_player_Y6  away_player_Y7  away_player_Y8
##  Min.   :3.000   Min.   : 3.00   Min.   : 3.00   Min.   : 3.000
##  1st Qu.:3.000   1st Qu.: 6.00   1st Qu.: 6.00   1st Qu.: 7.000
##  Median :3.000   Median : 7.00   Median : 7.00   Median : 7.000
##  Mean   :3.245   Mean   : 6.47   Mean   : 6.68   Mean   : 7.246
##  3rd Qu.:3.000   3rd Qu.: 7.00   3rd Qu.: 7.00   3rd Qu.: 8.000
##  Max.   :9.000   Max.   :10.00   Max.   :10.00   Max.   :10.000
##  NA's   :1832    NA's   :1832    NA's   :1832    NA's   :1832
##  away_player_Y9   away_player_Y10  away_player_Y11 home_player_1
##  Min.   : 5.000   Min.   : 6.000   Min.   : 7.00   Min.   :  2984
##  1st Qu.: 7.000   1st Qu.: 8.000   1st Qu.:10.00   1st Qu.: 30602
##  Median : 8.000   Median :10.000   Median :10.00   Median : 38230
##  Mean   : 8.022   Mean   : 9.161   Mean   :10.46   Mean   : 76638
##  3rd Qu.: 8.000   3rd Qu.:10.000   3rd Qu.:11.00   3rd Qu.: 96836
##  Max.   :11.000   Max.   :11.000   Max.   :11.00   Max.   :698273
##  NA's   :1833     NA's   :1833     NA's   :1839    NA's   :1224
##  home_player_2    home_player_3    home_player_4    home_player_5
##  Min.   :  2802   Min.   :  2752   Min.   :  2752   Min.   :  2752
##  1st Qu.: 32574   1st Qu.: 30602   1st Qu.: 30627   1st Qu.: 33579
##  Median : 42388   Median : 39731   Median : 41060   Median : 45996
##  Mean   :106854   Mean   : 91601   Mean   : 94540   Mean   :109528
##  3rd Qu.:159854   3rd Qu.:128037   3rd Qu.:145561   3rd Qu.:160243
##  Max.   :748432   Max.   :705484   Max.   :723037   Max.   :733787
##  NA's   :1315     NA's   :1281     NA's   :1323     NA's   :1316
##  home_player_6    home_player_7    home_player_8    home_player_9
##  Min.   :  2625   Min.   :  2625   Min.   :  2625   Min.   :  2625
##  1st Qu.: 31037   1st Qu.: 30895   1st Qu.: 32751   1st Qu.: 33332
##  Median : 41467   Median : 41432   Median : 43319   Median : 45605
##  Mean   :102309   Mean   : 97288   Mean   :107291   Mean   :111132
##  3rd Qu.:150944   3rd Qu.:141699   3rd Qu.:160243   3rd Qu.:164479
##  Max.   :750584   Max.   :692984   Max.   :693171   Max.   :730065
##  NA's   :1325     NA's   :1227     NA's   :1309     NA's   :1273
##  home_player_10   home_player_11   away_player_1    away_player_2
##  Min.   :  2625   Min.   :  2802   Min.   :  2796   Min.   :  2790
##  1st Qu.: 32465   1st Qu.: 32627   1st Qu.: 30622   1st Qu.: 32579
##  Median : 43296   Median : 42091   Median : 38289   Median : 42388
##  Mean   :105613   Mean   :103414   Mean   : 76628   Mean   :107615
##  3rd Qu.:158783   3rd Qu.:161291   3rd Qu.: 96836   3rd Qu.:159882
##  Max.   :742405   Max.   :726956   Max.   :698273   Max.   :748432
```

```
##  NA's   :1436     NA's   :1555     NA's   :1234     NA's   :1278
##  away_player_3    away_player_4    away_player_5    away_player_6
##  Min.   :  2752   Min.   :  2752   Min.   :  2790   Min.   :  2625
##  1st Qu.: 30464   1st Qu.: 30627   1st Qu.: 33454   1st Qu.: 31037
##  Median : 39892   Median : 41083   Median : 46212   Median : 41635
##  Mean   : 91127   Mean   : 95084   Mean   :109801   Mean   :102308
##  3rd Qu.:121080   3rd Qu.:145561   3rd Qu.:160844   3rd Qu.:151079
##  Max.   :705484   Max.   :728414   Max.   :746419   Max.   :722766
##  NA's   :1293     NA's   :1321     NA's   :1335     NA's   :1313
##  away_player_7    away_player_8    away_player_9    away_player_10
##  Min.   :  2625   Min.   :  2625   Min.   :  2625   Min.   :  2770
##  1st Qu.: 30920   1st Qu.: 32863   1st Qu.: 33435   1st Qu.: 32627
##  Median : 41433   Median : 45816   Median : 45860   Median : 45358
##  Mean   : 97898   Mean   :109265   Mean   :111087   Mean   :107149
##  3rd Qu.:144996   3rd Qu.:163612   3rd Qu.:164209   3rd Qu.:161291
##  Max.   :750435   Max.   :717248   Max.   :722766   Max.   :722766
##  NA's   :1235     NA's   :1341     NA's   :1328     NA's   :1441
##  away_player_11       goal              shoton             shotoff
##  Min.   :  2802   Length:25979       Length:25979       Length:25979
##  1st Qu.: 32747   Class :character   Class :character   Class :character
##  Median : 42652   Mode  :character   Mode  :character   Mode  :character
##  Mean   :104933
##  3rd Qu.:161660
##  Max.   :726956
##  NA's   :1554
##   foulcommit            card              cross
##  Length:25979       Length:25979       Length:25979
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     corner           possession            B365H            B365D
##  Length:25979       Length:25979       Min.   : 1.040   Min.   : 1.40
##  Class :character   Class :character   1st Qu.: 1.670   1st Qu.: 3.30
##  Mode  :character   Mode  :character   Median : 2.100   Median : 3.50
##                                        Mean   : 2.629   Mean   : 3.84
##                                        3rd Qu.: 2.800   3rd Qu.: 4.00
##                                        Max.   :26.000   Max.   :17.00
##                                        NA's   :3387     NA's   :3387
##      B365A             BWH              BWD              BWA
##  Min.   : 1.080   Min.   : 1.030   Min.   : 1.650   Min.   : 1.100
##  1st Qu.: 2.500   1st Qu.: 1.650   1st Qu.: 3.200   1st Qu.: 2.500
##  Median : 3.500   Median : 2.100   Median : 3.400   Median : 3.400
##  Mean   : 4.662   Mean   : 2.559   Mean   : 3.748   Mean   : 4.397
##  3rd Qu.: 5.250   3rd Qu.: 2.750   3rd Qu.: 3.800   3rd Qu.: 5.000
##  Max.   :51.000   Max.   :34.000   Max.   :19.500   Max.   :51.000
##  NA's   :3387     NA's   :3404     NA's   :3404     NA's   :3404
##       IWH              IWD              IWA              LBH
##  Min.   : 1.030   Min.   : 1.500   Min.   : 1.100   Min.   : 1.040
##  1st Qu.: 1.650   1st Qu.: 3.200   1st Qu.: 2.500   1st Qu.: 1.670
##  Median : 2.100   Median : 3.300   Median : 3.300   Median : 2.100
##  Mean   : 2.468   Mean   : 3.609   Mean   : 4.151   Mean   : 2.536
```

```
##  3rd Qu.: 2.600   3rd Qu.: 3.700   3rd Qu.: 4.600   3rd Qu.: 2.700  
##  Max.   :20.000   Max.   :11.000   Max.   :25.000   Max.   :26.000  
##  NA's   :3459     NA's   :3459     NA's   :3459     NA's   :3423    
##       LBD              LBA              PSH              PSD        
##  Min.   : 1.400   Min.   : 1.100   Min.   : 1.040   Min.   : 2.200  
##  1st Qu.: 3.200   1st Qu.: 2.500   1st Qu.: 1.720   1st Qu.: 3.410  
##  Median : 3.400   Median : 3.300   Median : 2.200   Median : 3.640  
##  Mean   : 3.712   Mean   : 4.385   Mean   : 2.816   Mean   : 4.132  
##  3rd Qu.: 3.750   3rd Qu.: 5.000   3rd Qu.: 2.980   3rd Qu.: 4.230  
##  Max.   :19.000   Max.   :51.000   Max.   :36.000   Max.   :29.000  
##  NA's   :3423     NA's   :3423     NA's   :14811    NA's   :14811   
##       PSA              WHH              WHD              WHA        
##  Min.   : 1.090   Min.   : 1.020   Min.   : 1.020   Min.   : 1.080  
##  1st Qu.: 2.560   1st Qu.: 1.670   1st Qu.: 3.200   1st Qu.: 2.500  
##  Median : 3.610   Median : 2.150   Median : 3.300   Median : 3.400  
##  Mean   : 4.973   Mean   : 2.579   Mean   : 3.665   Mean   : 4.483  
##  3rd Qu.: 5.410   3rd Qu.: 2.750   3rd Qu.: 3.750   3rd Qu.: 5.000  
##  Max.   :47.500   Max.   :26.000   Max.   :17.000   Max.   :51.000  
##  NA's   :14811    NA's   :3408     NA's   :3408     NA's   :3408    
##       SJH              SJD              SJA              VCH        
##  Min.   : 1.040   Min.   : 1.400   Min.   : 1.100   Min.   : 1.030  
##  1st Qu.: 1.670   1st Qu.: 3.250   1st Qu.: 2.500   1st Qu.: 1.700  
##  Median : 2.100   Median : 3.400   Median : 3.500   Median : 2.150  
##  Mean   : 2.566   Mean   : 3.756   Mean   : 4.622   Mean   : 2.668  
##  3rd Qu.: 2.750   3rd Qu.: 3.800   3rd Qu.: 5.250   3rd Qu.: 2.800  
##  Max.   :23.000   Max.   :15.000   Max.   :41.000   Max.   :36.000  
##  NA's   :8882     NA's   :8882     NA's   :8882     NA's   :3411    
##       VCD              VCA             GBH              GBD        
##  Min.   : 1.620   Min.   : 1.08   Min.   : 1.050   Min.   : 1.450  
##  1st Qu.: 3.300   1st Qu.: 2.55   1st Qu.: 1.670   1st Qu.: 3.200  
##  Median : 3.500   Median : 3.50   Median : 2.100   Median : 3.300  
##  Mean   : 3.899   Mean   : 4.84   Mean   : 2.499   Mean   : 3.648  
##  3rd Qu.: 4.000   3rd Qu.: 5.40   3rd Qu.: 2.650   3rd Qu.: 3.750  
##  Max.   :26.000   Max.   :67.00   Max.   :21.000   Max.   :11.000  
##  NA's   :3411     NA's   :3411    NA's   :11817    NA's   :11817   
##       GBA              BSH              BSD              BSA        
##  Min.   : 1.120   Min.   : 1.040   Min.   : 1.330   Min.   : 1.120  
##  1st Qu.: 2.500   1st Qu.: 1.670   1st Qu.: 3.250   1st Qu.: 2.500  
##  Median : 3.400   Median : 2.100   Median : 3.400   Median : 3.400  
##  Mean   : 4.353   Mean   : 2.498   Mean   : 3.661   Mean   : 4.406  
##  3rd Qu.: 5.000   3rd Qu.: 2.620   3rd Qu.: 3.750   3rd Qu.: 5.000  
##  Max.   :34.000   Max.   :17.000   Max.   :13.000   Max.   :34.000  
##  NA's   :11817    NA's   :11818    NA's   :11818    NA's   :11818   
```

```r
summary(player_atts)
```

```
##        id          player_fifa_api_id player_api_id        date          
##  Min.   :     1   Min.   :     2     Min.   :  2625   Length:183978     
##  1st Qu.: 45995   1st Qu.:155798     1st Qu.: 34763   Class :character  
##  Median : 91990   Median :183488     Median : 77741   Mode  :character  
##  Mean   : 91990   Mean   :165672     Mean   :135901                     
##  3rd Qu.:137984   3rd Qu.:199848     3rd Qu.:191080                     
##  Max.   :183978   Max.   :234141     Max.   :750584                     
##                                                                         
##  overall_rating   potential     preferred_foot     attacking_work_rate
```

```
##  Min.   :33.0   Min.   :39.00   Length:183978      Length:183978     
##  1st Qu.:64.0   1st Qu.:69.00   Class :character   Class :character  
##  Median :69.0   Median :74.00   Mode  :character   Mode  :character  
##  Mean   :68.6   Mean   :73.46                                        
##  3rd Qu.:73.0   3rd Qu.:78.00                                        
##  Max.   :94.0   Max.   :97.00                                        
##  NA's   :836    NA's   :836                                          
##  defensive_work_rate    crossing       finishing     heading_accuracy
##  Length:183978       Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   
##  Class :character    1st Qu.:45.00   1st Qu.:34.00   1st Qu.:49.00   
##  Mode  :character    Median :59.00   Median :53.00   Median :60.00   
##                      Mean   :55.09   Mean   :49.92   Mean   :57.27   
##                      3rd Qu.:68.00   3rd Qu.:65.00   3rd Qu.:68.00   
##                      Max.   :95.00   Max.   :97.00   Max.   :98.00   
##                      NA's   :836     NA's   :836     NA's   :836     
##  short_passing      volleys        dribbling         curve      
##  Min.   : 3.00   Min.   : 1.00   Min.   : 1.00   Min.   : 2.00  
##  1st Qu.:57.00   1st Qu.:35.00   1st Qu.:52.00   1st Qu.:41.00  
##  Median :65.00   Median :52.00   Median :64.00   Median :56.00  
##  Mean   :62.43   Mean   :49.47   Mean   :59.18   Mean   :52.97  
##  3rd Qu.:72.00   3rd Qu.:64.00   3rd Qu.:72.00   3rd Qu.:67.00  
##  Max.   :97.00   Max.   :93.00   Max.   :97.00   Max.   :94.00  
##  NA's   :836     NA's   :2713    NA's   :836     NA's   :2713   
##  free_kick_accuracy  long_passing    ball_control    acceleration  
##  Min.   : 1.00      Min.   : 3.00   Min.   : 5.00   Min.   :10.00  
##  1st Qu.:36.00      1st Qu.:49.00   1st Qu.:58.00   1st Qu.:61.00  
##  Median :50.00      Median :59.00   Median :67.00   Median :69.00  
##  Mean   :49.38      Mean   :57.07   Mean   :63.39   Mean   :67.66  
##  3rd Qu.:63.00      3rd Qu.:67.00   3rd Qu.:73.00   3rd Qu.:77.00  
##  Max.   :97.00      Max.   :97.00   Max.   :97.00   Max.   :97.00  
##  NA's   :836        NA's   :836     NA's   :836     NA's   :836    
##   sprint_speed      agility        reactions       balance     
##  Min.   :12.00   Min.   :11.00   Min.   :17.0   Min.   :12.00  
##  1st Qu.:62.00   1st Qu.:58.00   1st Qu.:61.0   1st Qu.:58.00  
##  Median :69.00   Median :68.00   Median :67.0   Median :67.00  
##  Mean   :68.05   Mean   :65.97   Mean   :66.1   Mean   :65.19  
##  3rd Qu.:77.00   3rd Qu.:75.00   3rd Qu.:72.0   3rd Qu.:74.00  
##  Max.   :97.00   Max.   :96.00   Max.   :96.0   Max.   :96.00  
##  NA's   :836     NA's   :2713    NA's   :836    NA's   :2713   
##    shot_power       jumping         stamina         strength    
##  Min.   : 2.00   Min.   :14.00   Min.   :10.00   Min.   :10.00  
##  1st Qu.:54.00   1st Qu.:60.00   1st Qu.:61.00   1st Qu.:60.00  
##  Median :65.00   Median :68.00   Median :69.00   Median :69.00  
##  Mean   :61.81   Mean   :66.97   Mean   :67.04   Mean   :67.42  
##  3rd Qu.:73.00   3rd Qu.:74.00   3rd Qu.:76.00   3rd Qu.:76.00  
##  Max.   :97.00   Max.   :96.00   Max.   :96.00   Max.   :96.00  
##  NA's   :836     NA's   :2713    NA's   :836     NA's   :836    
##    long_shots      aggression    interceptions    positioning   
##  Min.   : 1.00   Min.   : 6.00   Min.   : 1.00   Min.   : 2.00  
##  1st Qu.:41.00   1st Qu.:51.00   1st Qu.:34.00   1st Qu.:45.00  
##  Median :58.00   Median :64.00   Median :57.00   Median :60.00  
##  Mean   :53.34   Mean   :60.95   Mean   :52.01   Mean   :55.79  
##  3rd Qu.:67.00   3rd Qu.:73.00   3rd Qu.:68.00   3rd Qu.:69.00  
##  Max.   :96.00   Max.   :97.00   Max.   :96.00   Max.   :96.00  
```

```
##  NA's   :836     NA's   :836     NA's   :836     NA's   :836
##      vision        penalties      marking      standing_tackle
##  Min.   : 1.00   Min.   : 2    Min.   : 1.00   Min.   : 1.00
##  1st Qu.:49.00   1st Qu.:45    1st Qu.:25.00   1st Qu.:29.00
##  Median :60.00   Median :57    Median :50.00   Median :56.00
##  Mean   :57.87   Mean   :55    Mean   :46.77   Mean   :50.35
##  3rd Qu.:69.00   3rd Qu.:67    3rd Qu.:66.00   3rd Qu.:69.00
##  Max.   :97.00   Max.   :96    Max.   :96.00   Max.   :95.00
##  NA's   :2713    NA's   :836   NA's   :836     NA's   :836
##  sliding_tackle   gk_diving     gk_handling     gk_kicking
##  Min.   : 2     Min.   : 1.0   Min.   : 1.00   Min.   : 1
##  1st Qu.:25     1st Qu.: 7.0   1st Qu.: 8.00   1st Qu.: 8
##  Median :53     Median :10.0   Median :11.00   Median :12
##  Mean   :48     Mean   :14.7   Mean   :16.06   Mean   :21
##  3rd Qu.:67     3rd Qu.:13.0   3rd Qu.:15.00   3rd Qu.:15
##  Max.   :95     Max.   :94.0   Max.   :93.00   Max.   :97
##  NA's   :2713   NA's   :836    NA's   :836     NA's   :836
##  gk_positioning   gk_reflexes
##  Min.   : 1.00   Min.   : 1.00
##  1st Qu.: 8.00   1st Qu.: 8.00
##  Median :11.00   Median :11.00
##  Mean   :16.13   Mean   :16.44
##  3rd Qu.:15.00   3rd Qu.:15.00
##  Max.   :96.00   Max.   :96.00
##  NA's   :836     NA's   :836
```