Capestone-Final Project-English premier league results prediction model

Jersson Placido

16/04/2021

1. INTRODUCTION

There is a science to international football predictions and any true and seasoned fan knows well that to optimize your chances of winning bets, certain variables must be taken into account. Observations such as goal spreads, possession, shots on goal and the location of shots on goal by a certain player or team as a whole are indicators to take into account when looking to make a good bet.

The quality of the team is not just about the talented players they have, it is also about the unity of the team and how they work together, the compatibility of the styles of the players with each other and the quality of the club director. The style of play of both the player as an individual and the team as a unit is another important determining factor to consider when making soccer predictions. A deep understanding of the game and the quality time spent observing the players in action provide relevant information that could skew results in favor of the best.

As many variables can affect the result of a football match and player characteristics and virtues can reflect in the final result of the match. In this report, I utilized the players' ratings from the FIFA 2020 video game as predictors for the results of the English premier league matches of the 2020 season. To achieve this, I joined two data sets one including the results of the premier league from 1993-94 season until the current 2020-21 season, and one including the players' ratings for the FIFA 2020 video game. During the whole report I utilize three data sets, one using the away team's players' ratings, another the home team's players' ratings, and finally one using both player ratings.

The report includes an initial data bases exploration and the model development section, comments and conclusions.

2.METHODS

2.1 DATABASES DOWNLOAD

To develop a prediction system for the results of the English Premier League (EPL) using the player ratings included in the FIFA video game. I obtained the data bases form two different websites. the EPL matches results were collected and stored in the next web site:

https://www.kaggle.com/irkaal/english-premier-league-results

while, the EPL FIFA 2020 videogame ratings were collected and stored in the next web site:

https://www.kaggle.com/aricht1995/premier-league-epl-player-information.

The libraries required for developing the EPL results predictor based on the FIFA 2020 videogame ratings are next:

```
library(tidyverse)
library(caret)
library(data.table)
library(lubridate)
library (readr)
library(corrplot)
library(Rborist)
```

The following code was used to download the csv files directly from my github account:

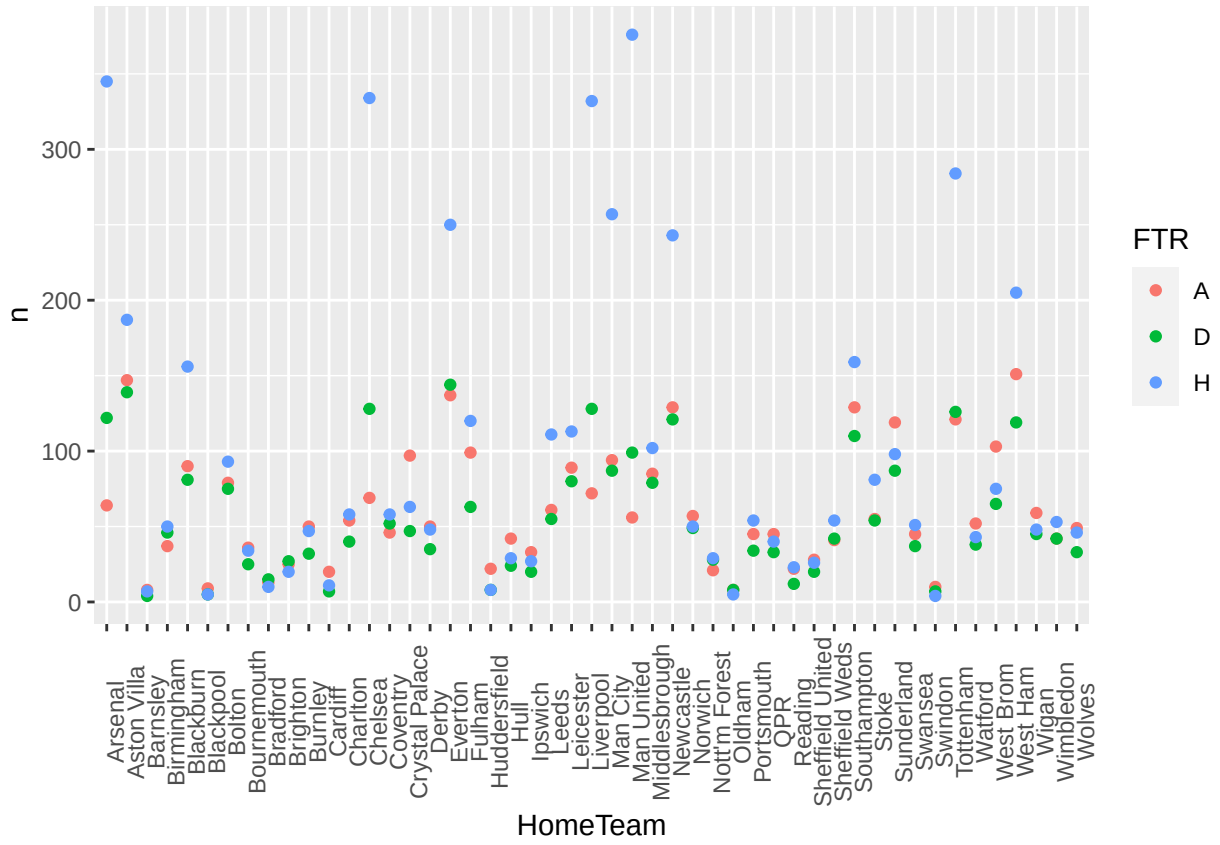
```
EPL_results <- read_csv("https://raw.githubusercontent.com/jepeteso/finalproject/main/results.csv",
         col_types = list(col_character(), col_character(),  col_character(), col_character(),
                          col_number(), col_number(), col_character(), col_number(),
                          col_number(), col_character(), col_character(),  col_number(),
                          col_number(), col_number(), col_number(), col_number(),
                          col_number(), col_number(), col_number(),col_number(),
                          col_number(),col_number(),col_number()))

urlfile="https://raw.githubusercontent.com/jepeteso/finalproject/main/EPL%20Player%20Info%20fifa.csv"
EPL_fifa_ratings<- read_csv(url(urlfile))
```

2.2 DATA SETS EXPLORATION

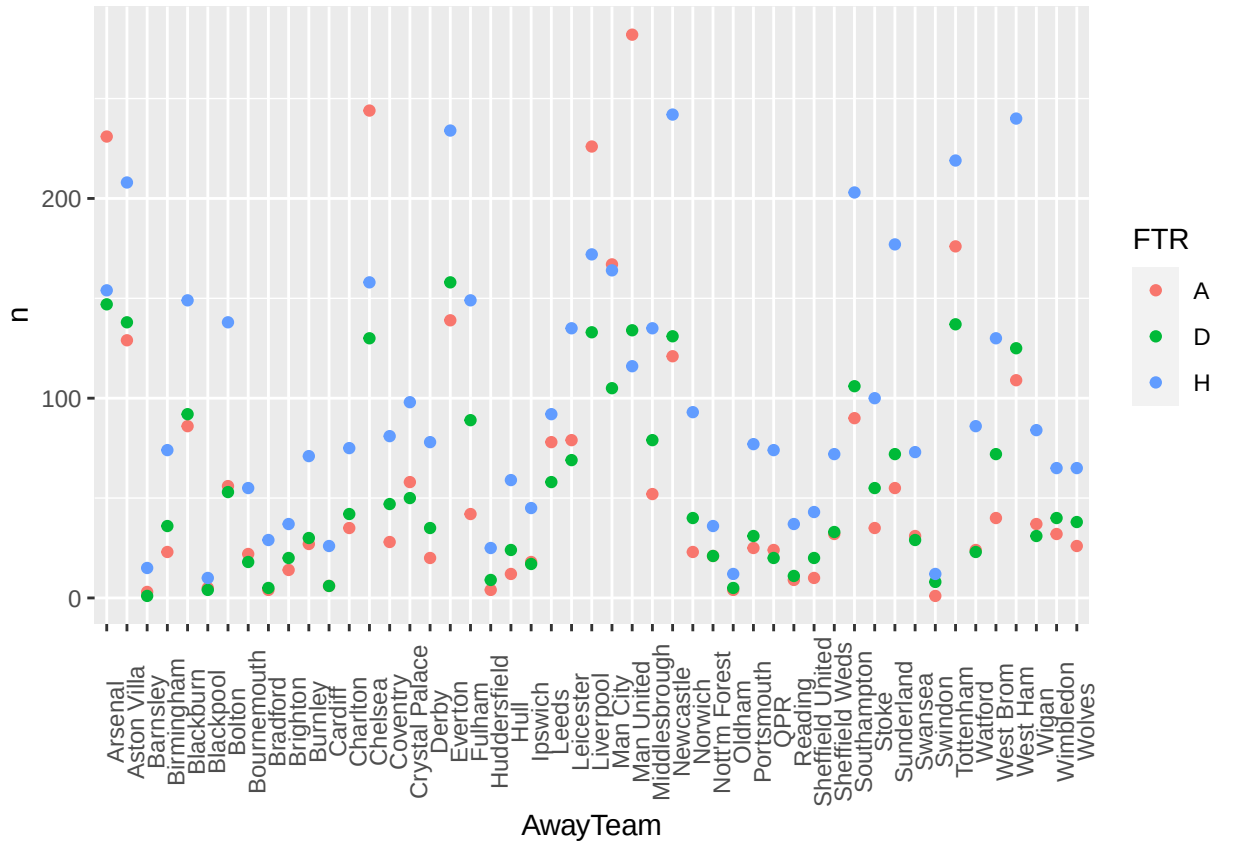
2.2.1 EPL results data set exploration

To evaluate the EPL results data set, I wanted to check how each team has performed as a home and away team by computing the final time results in terms of wins by the home team (H), wins by the away team (A) and draws (D). This analysis was generated using the following code

```
EPL_results %>% group_by(HomeTeam, FTR) %>% summarise(n=n()) %>% ggplot(aes(HomeTeam,n, color=FTR))+
  geom_point() + theme(axis.text.x = element_text(angle = 90))
```

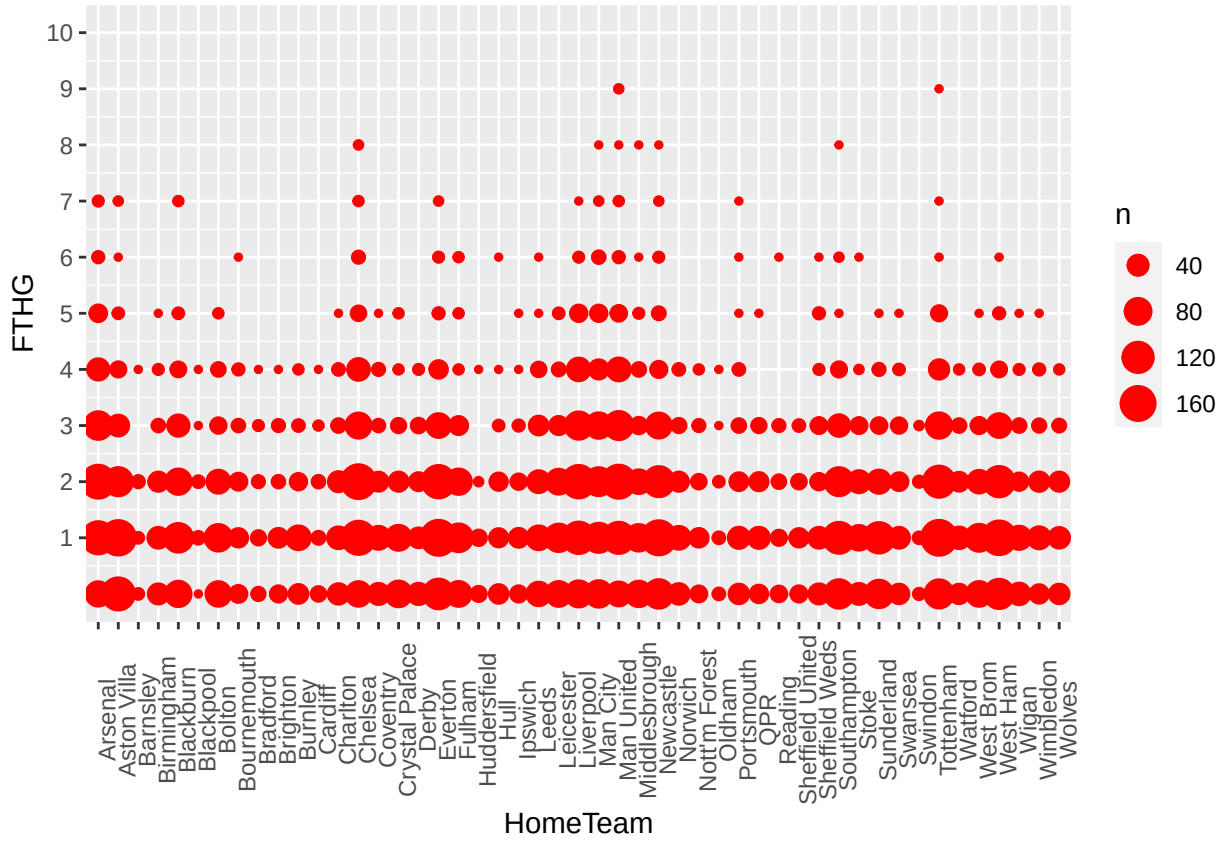



```
EPL_results %>% group_by(AwayTeam, FTR) %>% summarise(n=n()) %>% ggplot(aes(AwayTeam,n, color=FTR))+
  geom_point() + theme(axis.text.x = element_text(angle = 90))
```

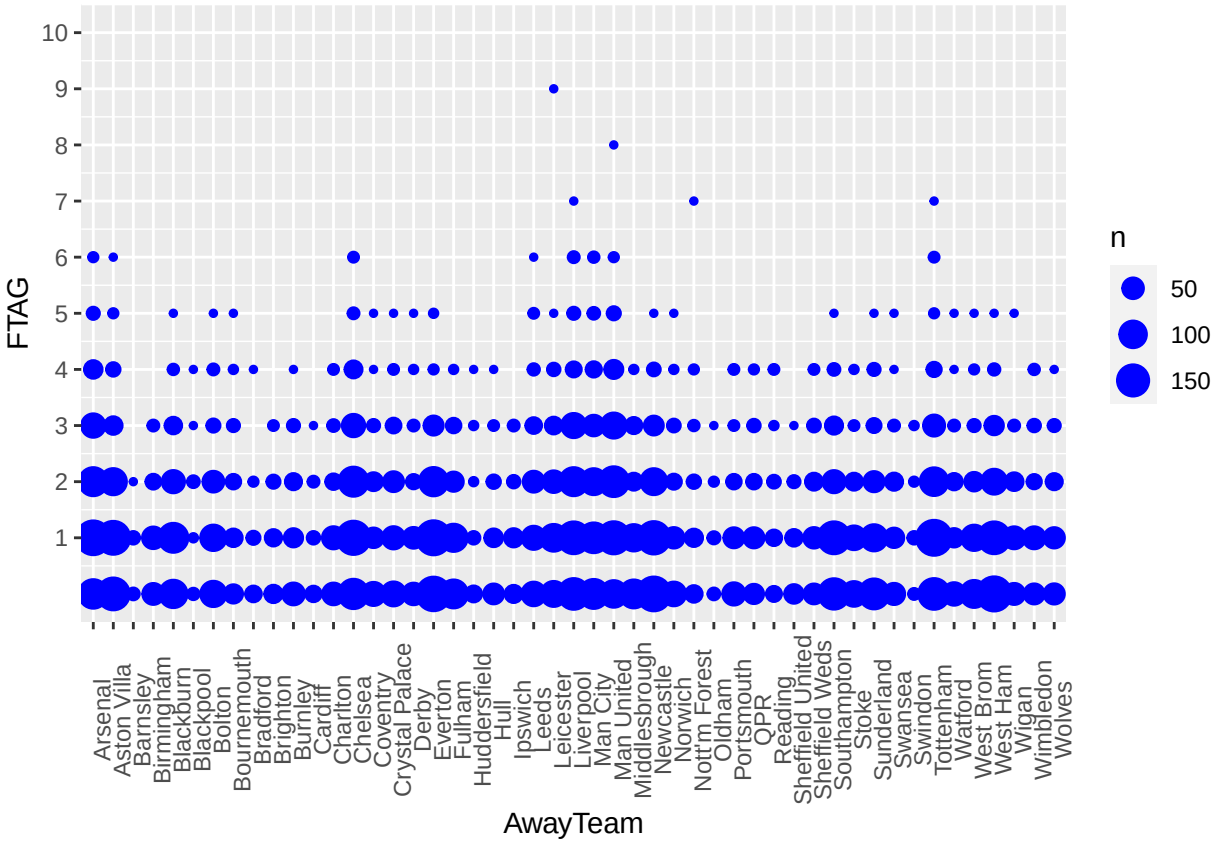


It is interesting to see how the teams behave differently as home team and away team, only four teams have a victory record as an away team, compared with 24 teams with winning record as a local team. As home and away indicate a difference in the full time results (FTR), next I summarized the goals scored by home and away teams and how many times each singular score have been obtained.

```
#amount of times of an specific amount of goals have been scored by an specific hometeam
EPL_results %>% group_by(HomeTeam, FTHG) %>% summarise(n=n()) %>% ggplot(aes(HomeTeam, FTHG, size=n)) +
  geom_point(color="red") +   theme(axis.text.x = element_text(angle = 90)) +
                                scale_y_continuous(limits=c(0,10), breaks=1:10)
```



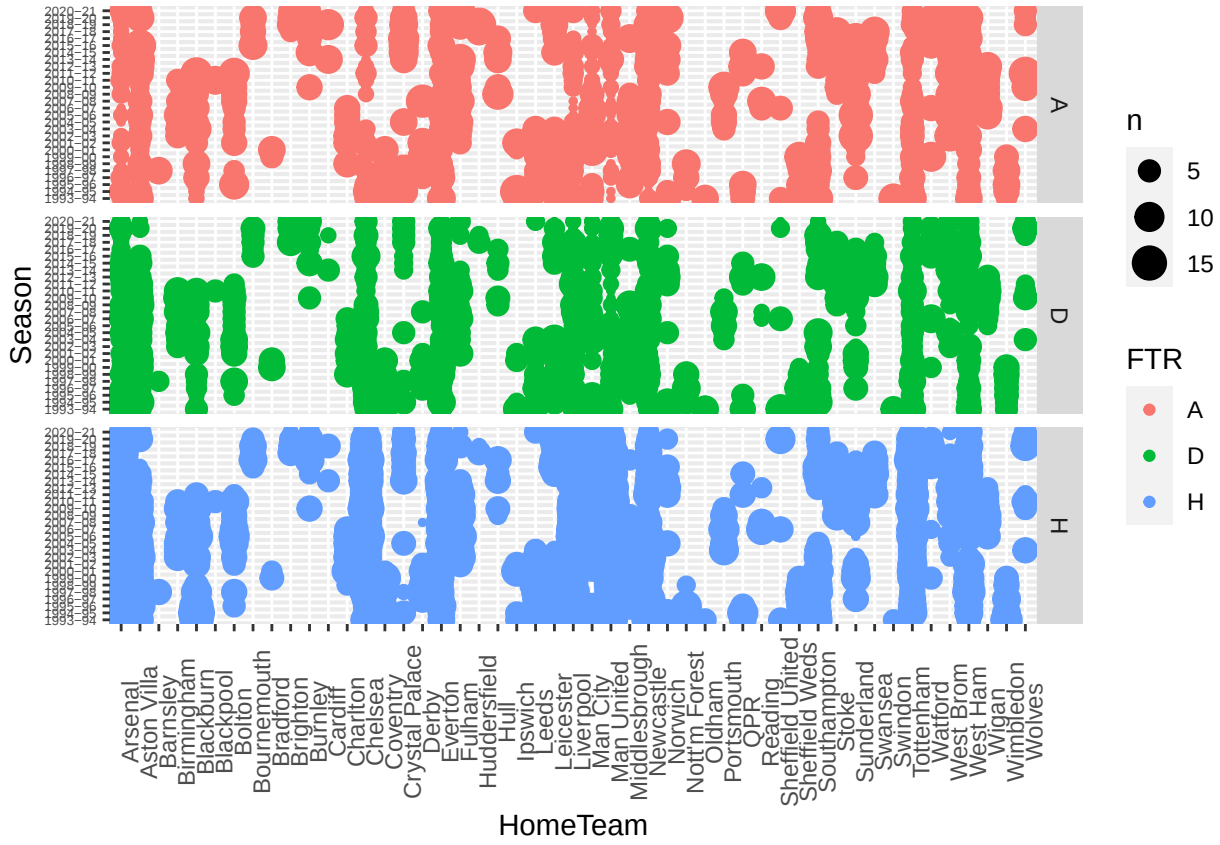

```
#amount of times of an specific amount of goals have been scored by an specific Awayteam
EPL_results %>% group_by(AwayTeam, FTAG) %>% summarise(n=n()) %>% ggplot(aes(AwayTeam, FTAG, size=n)) +
  geom_point(color="Blue") +   theme(axis.text.x = element_text(angle = 90)) +
  scale_y_continuous(limits=c(0,10), breaks=1:10)
```



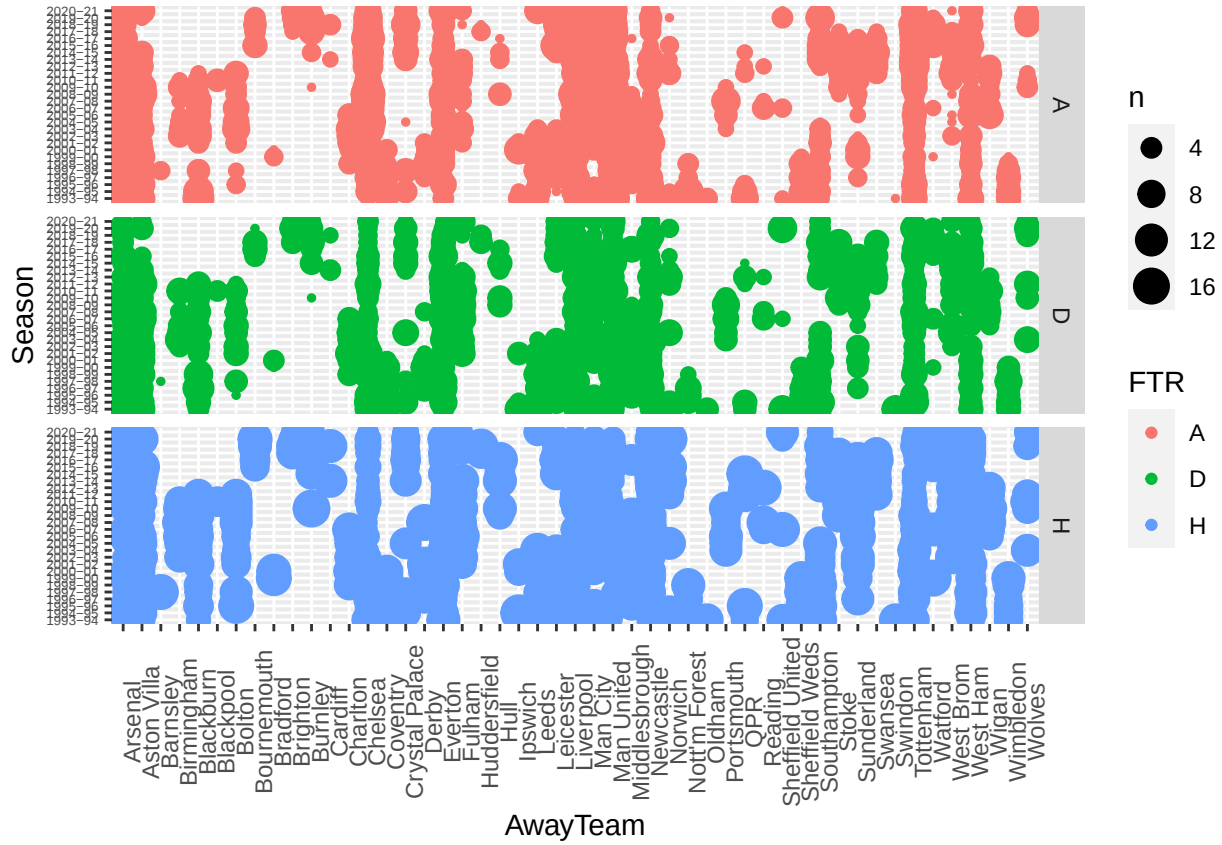
the past plots indicated a difference between full time away team goals (FTAG) and full time home team goals (FTHG). As expected the home teams has scored more goal than the away teams. Additionally, It was possible to see that scores above 4 goals happened more commonly in home teams than away teams.

As the data base has information from the season 1993-1994, A summary including results as home and away teams during those years is included and plotted using the following code

```
# Results summary analyzing full time results, season and home team
EPL_results %>% group_by(Season, FTR, HomeTeam)%>% summarise(n=n())%>%
  ggplot(aes(HomeTeam, Season, color= FTR, size=n)) + geom_point() +facet_grid(FTR ~ .) +
  theme(axis.text.y = element_text(size=5), axis.text.x = element_text(angle = 90))
```




```
# Results summary analyzing full time results, season and away team
EPL_results %>% group_by(Season, FTR, AwayTeam)%>% summarise(n=n())%>%
  ggplot(aes(AwayTeam, Season, color= FTR, size=n)) + geom_point() +facet_grid(FTR ~ .) +
  theme(axis.text.y = element_text(size=5), axis.text.x = element_text(angle = 90))
```



2.2.2 Exploration of FIFA 20 video game ratings for EPL.

After an initial exploration I discovered that the FIFA ratings dataset had issues associated with the column names, first, the presence of spaces in the column names and second the presence of words between parenthesis. To remove these issues in the column titles I utilized the following code

```
#as several of the the column titles included spaces, it is necessary to replace the empty spaces
#with a character in this case I will utilize "_"

new_column_names <- str_replace_all(colnames(EPL_fifa_ratings), " ", "_")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names)
#colnames(EPL_fifa_ratings)

#the columns that have "(Euros)" in their names are removed using the following code
new_column_names2 <- str_replace_all(colnames(EPL_fifa_ratings), "_\\(Euros\\)", "")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names2)
#colnames(EPL_fifa_ratings)

#the columns that have "(FIFA)" in their names are removed using the following code
new_column_names3 <- str_replace_all(colnames(EPL_fifa_ratings), "\\(FIFA\\)", "")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names3)
#colnames(EPL_fifa_ratings)
```

```
#the columns that have "(Short)" in their names are removed using the following code
new_column_names4 <- str_replace_all(colnames(EPL_fifa_ratings), "_\\(short\\)", "S")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names4)

#the columns that have "(kg)" in their names are removed using the following code
new_column_names5 <- str_replace_all(colnames(EPL_fifa_ratings), "_\\(kg\\)", "")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names5)

#the columns that have "_" at the beginning of the name are removed using the following code
new_column_names6 <- str_replace_all(colnames(EPL_fifa_ratings), "_Dribbling", "Dribbling1")
old_column_names<- colnames(EPL_fifa_ratings)
EPL_fifa_ratings <- EPL_fifa_ratings %>% rename_at(vars(old_column_names), ~new_column_names6)
```

To facilitate the future work, the columns with high number of NAs were removed and the rows from the remaining columns were also removed using the following code

```
# proportion of NA per column
NAS <-colMeans(is.na(EPL_fifa_ratings))%>% data.frame()

#give a column name to the average
names(NAS)[1] <-"variable"

#identify the columns with a proportion of data above 90%
HighNA<- NAS %>% filter(variable > 0.10) %>% rownames()

#create a new data set without the columns with a proportion of data above 90% and
#removal rows with NA from the data set
EPL_fifa_ratings <- EPL_fifa_ratings%>% select(!HighNA) %>% drop_na()
```

Next, I summarized some player ratings and grouped them by club names, additionally, I included a new variable call money value to see how the cost of a player and their ratings overall are correlated.

```
Clubs_summaries <- EPL_fifa_ratings %>% group_by(Club_NameS) %>%
  mutate(MoneyValue=Market_Value/Overall) %>%
  summarise(Age=mean(Age), Weight=mean(Weight),Overall=mean(Overall),
            MarketValue=mean(Market_Value), Potential=mean(Potential), Wages=mean(FIFA_Wage),
            MoneyValue=mean(MoneyValue)) %>%   tibble()
```

```
Clubs_summaries %>% knitr::kable()
```

Table 1. EPL clubs facts

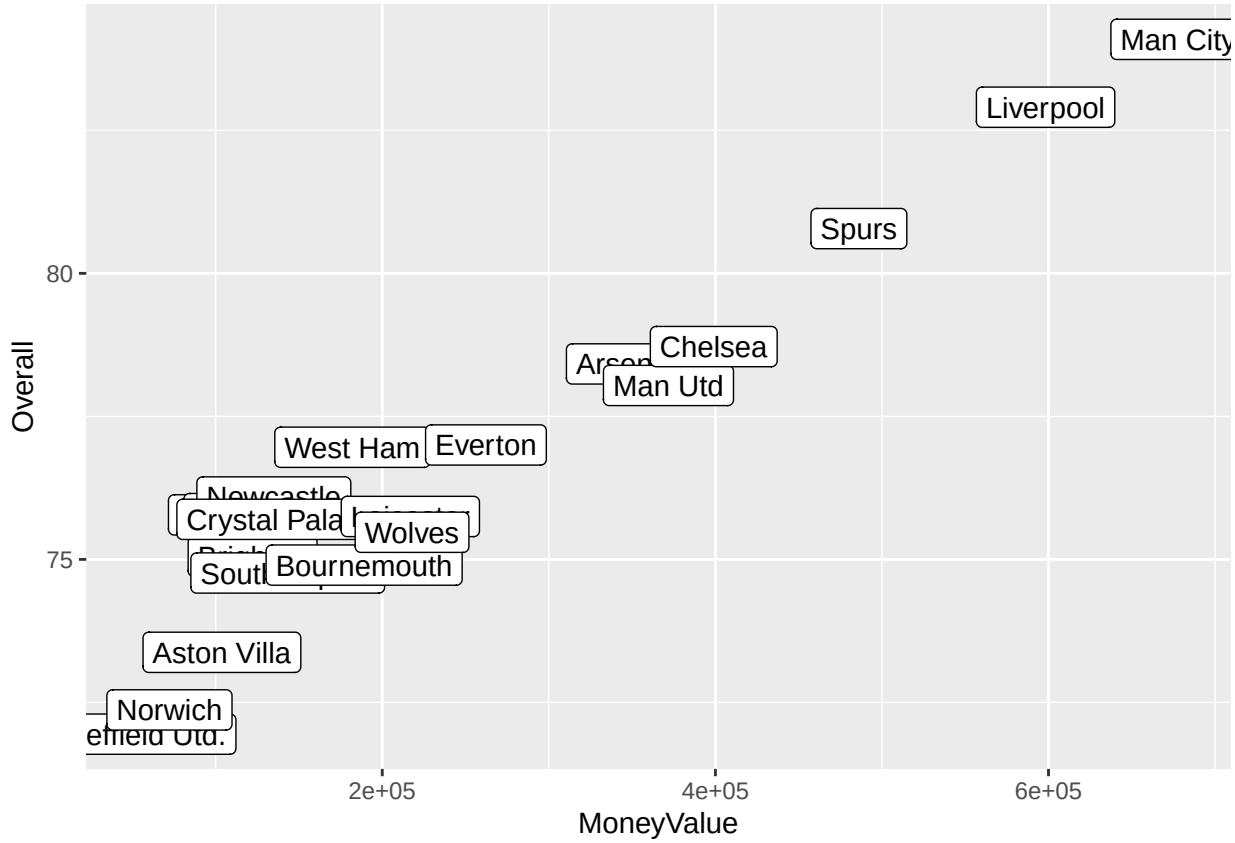
Club_NameS	Age	Weight	Overall	MarketValue	Potential	Wages	MoneyValue
Arsenal	25.41667	77.08333	78.41667	27979167	83.79167	76333.33	346181.97
Aston Villa	26.20833	79.75000	73.37500	7750000	77.54167	36083.33	103747.12
Bournemouth	25.30000	75.75000	74.90000	14500000	80.05000	45400.00	189116.72

Club_NameS	Age	Weight	Overall	MarketValue	Potential	Wages	MoneyValue
Brighton	26.70000	78.00000	75.05000	9312500	77.35000	38100.00	122163.00
Burnley	28.10000	78.25000	75.80000	8925000	77.60000	39800.00	116539.20
Chelsea	25.44000	79.08000	78.72000	32260000	83.92000	94400.00	399020.67
Crystal Palace	28.65000	77.85000	75.70000	10875000	76.45000	46450.00	139125.36
Everton	26.95833	76.41667	77.00000	20500000	80.33333	64000.00	262244.95
Leicester	26.33333	78.79167	75.75000	16802083	79.54167	55458.33	216870.96
Liverpool	27.28571	74.95238	82.90476	51047619	84.85714	119761.90	598237.57
Man City	26.59091	73.50000	84.09091	57977273	87.18182	173090.91	677957.61
Man Utd	25.64000	76.32000	78.04000	30130000	82.80000	95080.00	371826.97
Newcastle	27.36364	78.40909	76.09091	10409091	78.27273	35590.91	134907.02
Norwich	26.63158	76.26316	72.36842	5255263	75.05263	28526.32	72236.48
Sheffield Utd.	26.77778	76.77778	71.94444	3902778	75.05556	22222.22	53524.43
Southampton	25.57143	75.00000	74.76190	10904762	79.09524	38095.24	143253.44
Spurs	25.69565	78.73913	80.78261	40608696	84.52174	102826.09	486156.85
Watford	27.81481	80.11111	75.77778	8333333	78.33333	49000.00	107629.74
West Ham	28.12000	78.48000	76.96000	14430000	79.20000	67080.00	182088.78
Wolves	25.26316	77.78947	75.47368	16973684	80.63158	58842.11	217890.89

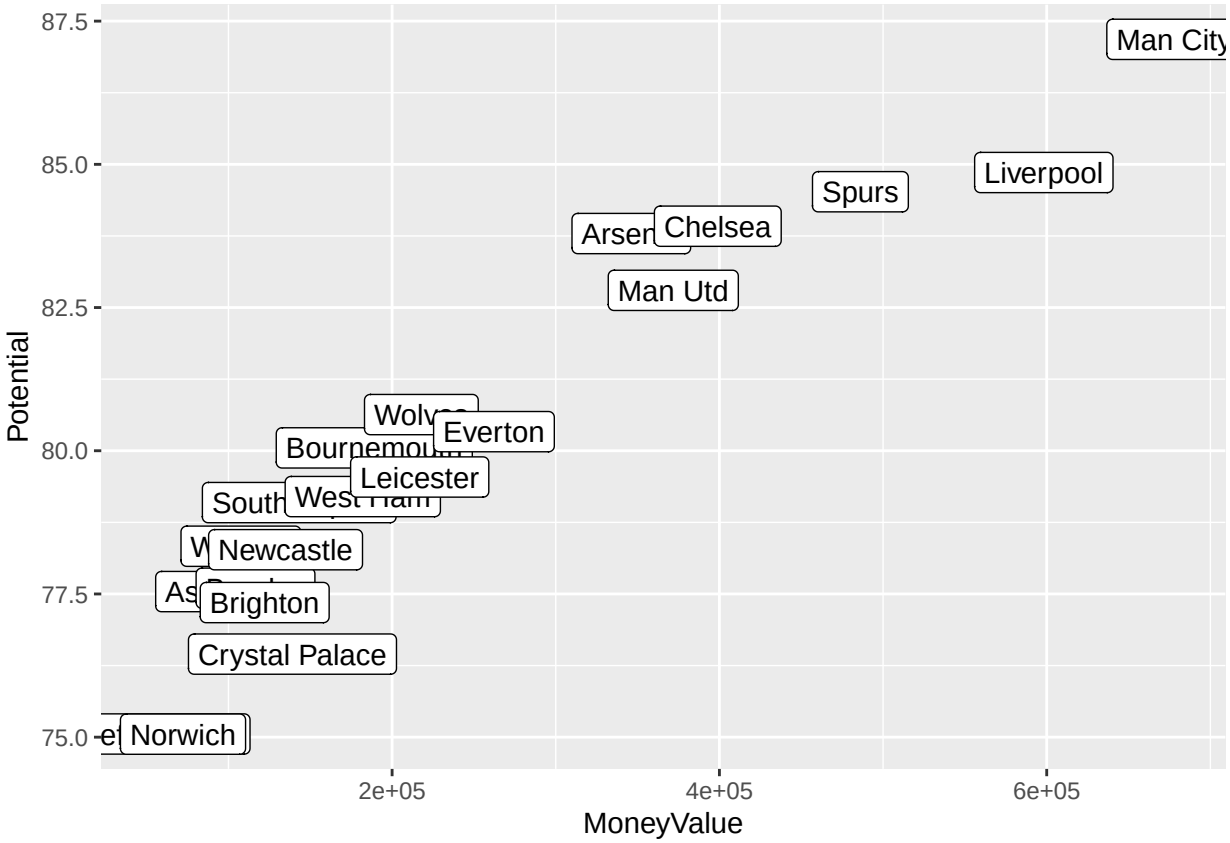
Crystal palace has the oldest team, while, wolves, have the youngest team. Watford is the heaviest team and Man city is the lightest team. Man city has the highest market value, overall ratings, potential, and wages. while Sheffield Utdhas the lowest values of those variables.

The following plots were developed to evaluate how the money value is associated with potential, overall and wages by team

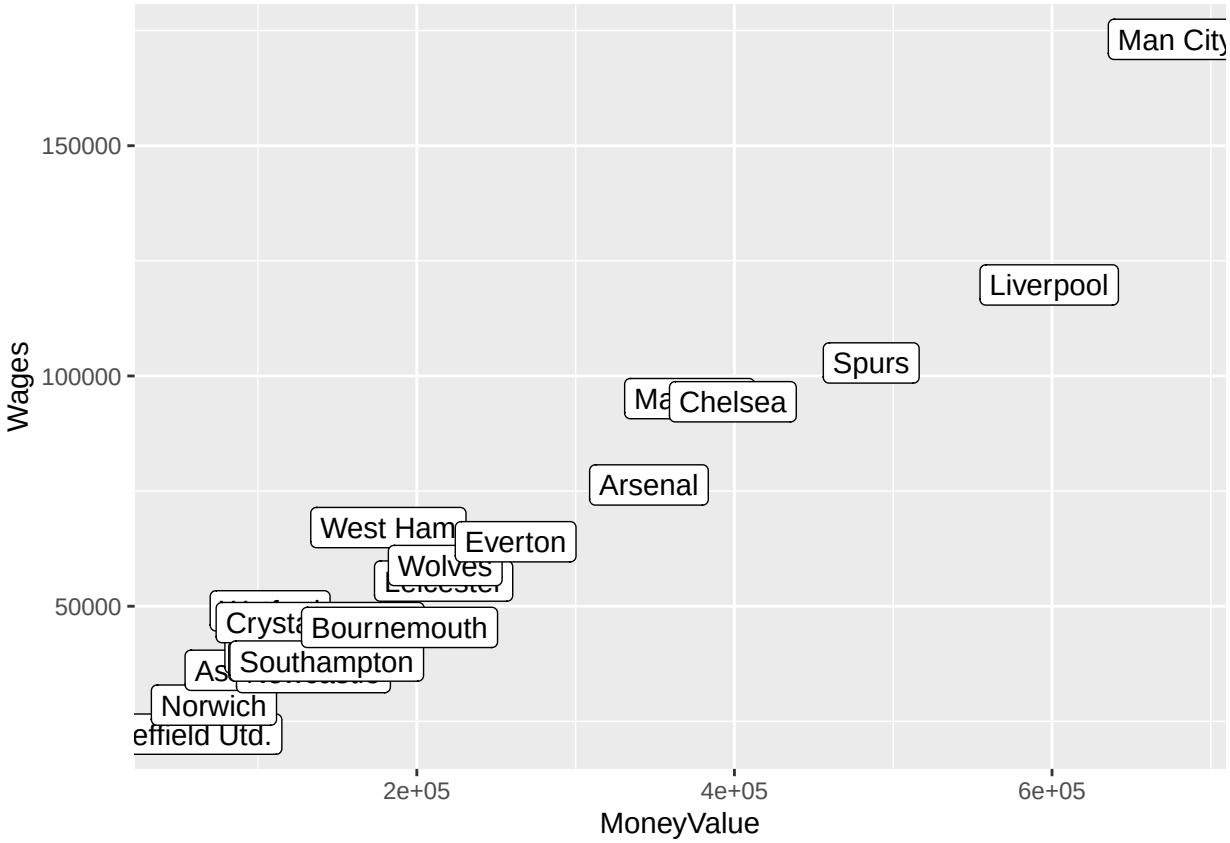
```
Clubs_summaries %>%  arrange(MoneyValue) %>%
  ggplot(aes(MoneyValue, Overall, label=Club_NameS)) + geom_label()
```




```
Clubs_summaries %>%  arrange(MoneyValue) %>%
  ggplot(aes(MoneyValue, Potential, label=Club_NameS)) + geom_label()
```



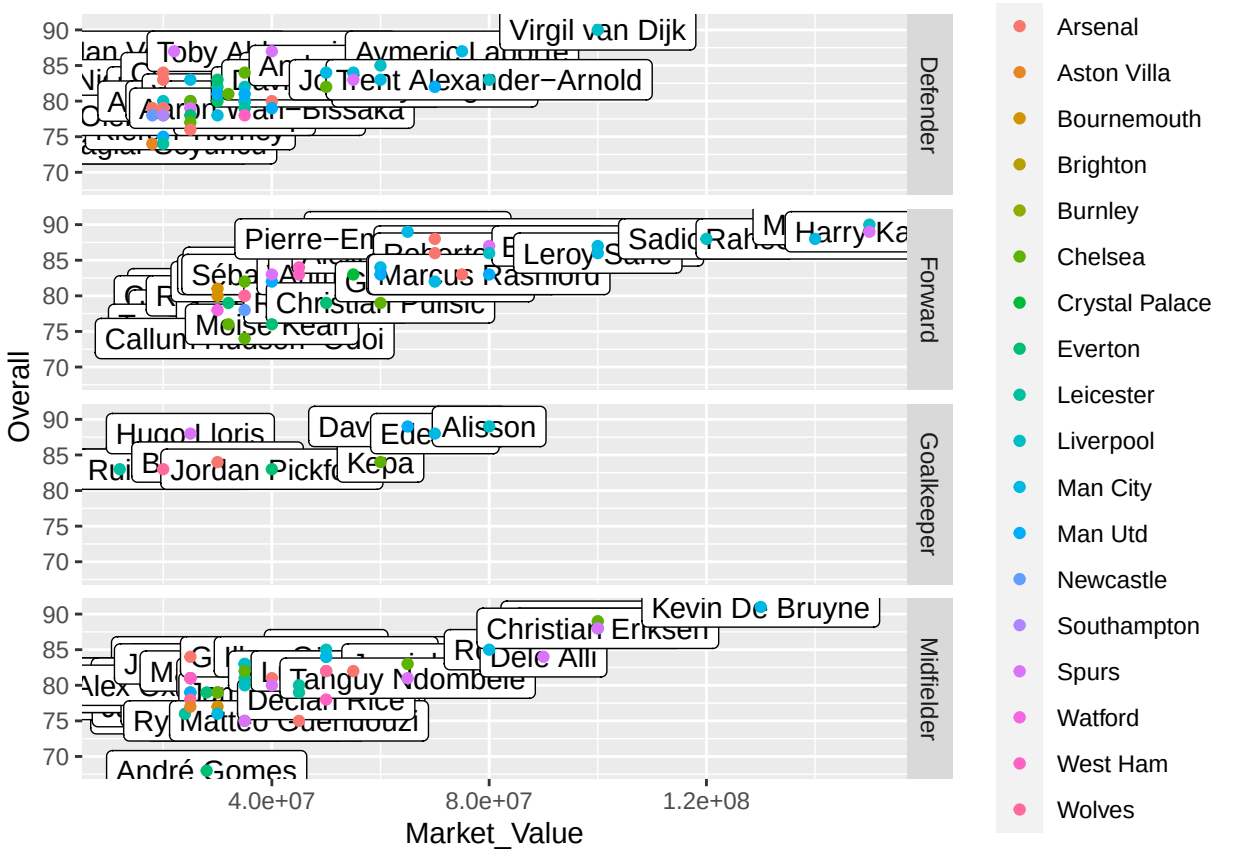

```
Clubs_summaries%>%  arrange(MoneyValue)  %>%
  ggplot(aes(MoneyValue, Wages, label=Club_NameS)) + geom_label()
```



The overall ratings and money value evidence 4 groups, one including liverpool, man city and spurs, the second including arsenal, chelsea and man Utd, a third group including norwich, sheffield and Aston villa. the other group include the rest of the teams. the potential showed a clear difference between the six most important teams and the rest of the EPL. The wages evidenced a significant difference between Man city and the rest of the teams.

Similarly, I analyzed how individual players overall and market value are associated.

```
# plots exhibiting player names and their market_value, overall, clubnames, wages and team
EPL_fifa_ratings%>% group_by(General_Position) %>%
  filter(Market_Value>mean(Market_Value)) %>%
  select(Player_Name, Market_Value, Club_NameS, Overall ) %>%
  arrange(Market_Value) %>% ggplot(aes(Market_Value, Overall, label=Player_Name, Color=Club_NameS))+
  geom_label()+ geom_point(aes(color=Club_NameS))+facet_grid(General_Position ~ .)
```



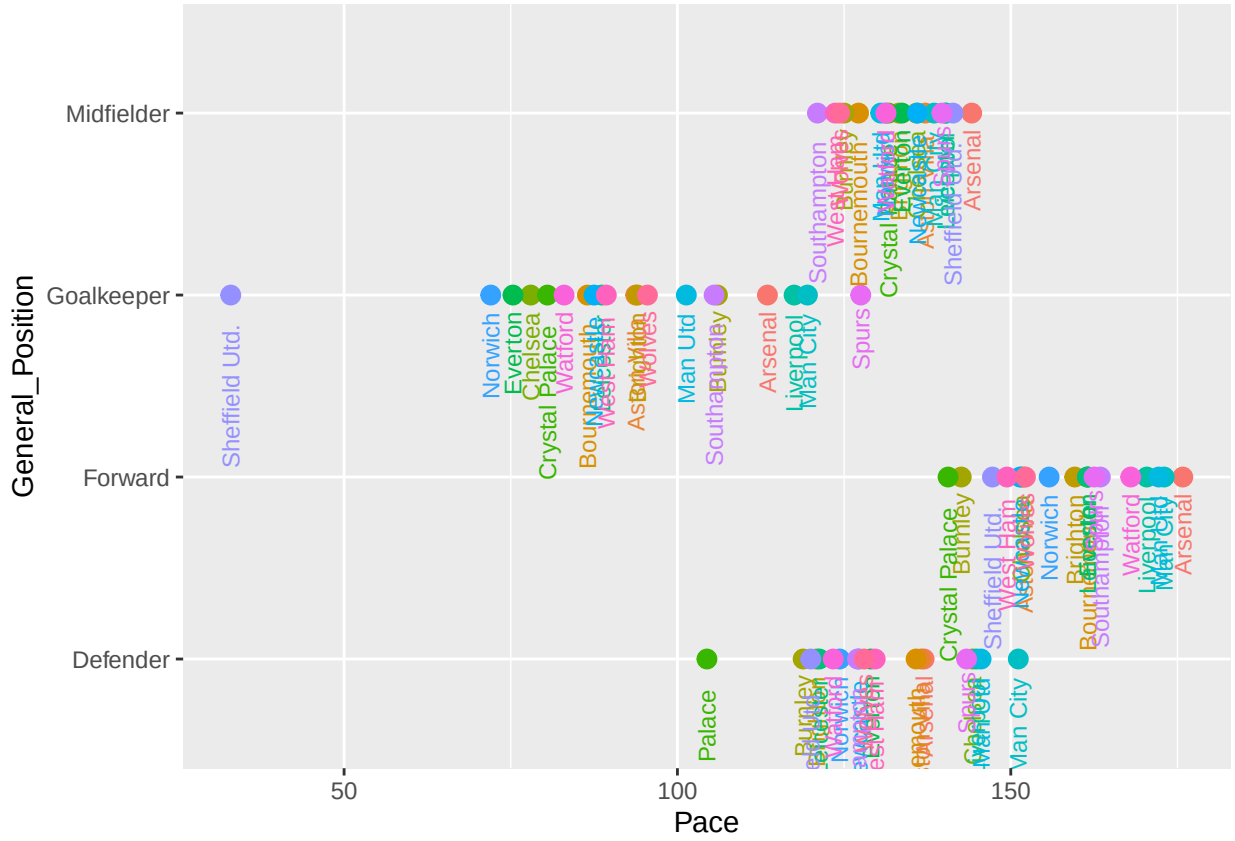
The overall rating and market value figure includes all the players above the average market value, the position with higher market value players are forwards with 4 players valued above 100 million, followed by midfielders and defenders.The players with high market value have high overall, however, some players with high overall rating do not have high market value, this is specially true in the forwards position, where several players have high overall rating and low or medium market value.

As the FIFA ratings describe several variables, I grouped several abilities in 7 mayor groups Pace Shooting Passing Dribbling2 Defending Physical Goalkeeping to facilitate their analysis.

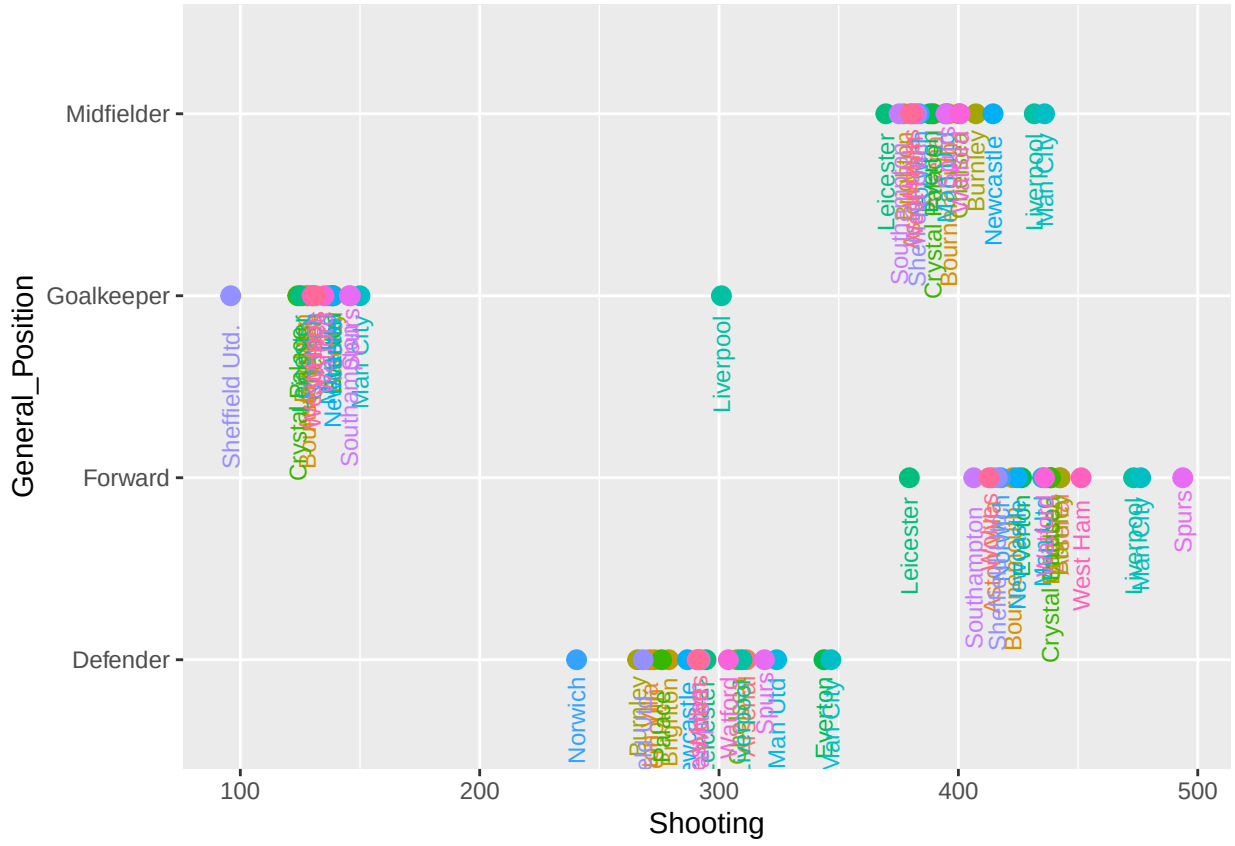
```
# summary of players attributes in 7 categories and grouped by club name
Ratingssummary <- EPL_fifa_ratings%>% group_by(Club_NameS, General_Position)%>%
  mutate(Pace=Acceleration+Sprint_Speed,
         Shooting= Attacking_Finishing+Long_Shots+Penalities+Positioning+Shot_Power+Attacking_Volleys,
         Passing=Attacking_Crossing + Curve + FK_Accuracy + Long_Passing + Attacking_Short_Passing + Vis
         Dribbling2= Agility + Balance + Ball_Control + Composure + Dribbling1 + Reactions,
         Defending= Attacking_Heading_Accuracy + Interceptions + Marking + Sliding_Tackle + Standing_Ta
         Physical= Aggression + Jumping + Stamina + Strength,
         Goalkeeping= GK_Diving2 + GK_Handling2 + GK_Kicking2 + GK_Positioning2 + GK_Reflexes2 )%>%
  select(Pace,Shooting, Passing, Dribbling2, Defending, Physical,Goalkeeping) %>%
  summarise(Pace=mean(Pace),
            Shooting=mean(Shooting),
            Passing=mean(Passing),
            Dribbling=mean(Dribbling2),
            Defending=mean(Defending),
            Physical=mean(Physical),
            Goalkeeping=mean(Goalkeeping)) %>% tibble()
```

The information contained in the summary was used to create plots describing the mayor attributes by team and position. The fastest forwards and midfielders were from arsenal while the fastest defender and forward were from Man city. the defensive pace was widely distributed while the midfielder were less distributed. In terms of shooting, it was possible to see two groups in each position, all of them included the Liverpool, however, the spurs had the best shooting forwards and the man city had the best shooting midfielders and defenders. It is significant that Man city and Liverpool are on top of almost every category while Sheffield Utd and Norwich are last in almost all.

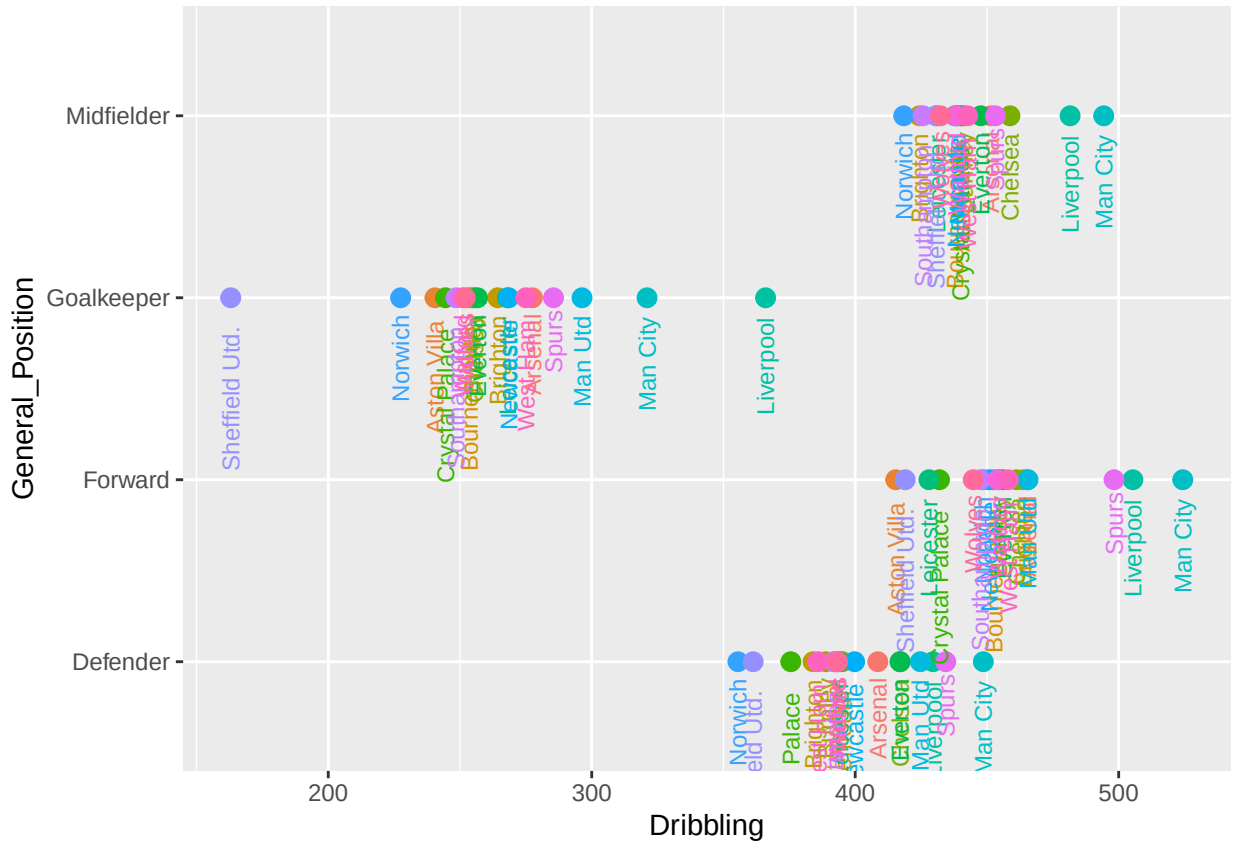
```
#pace plot
Ratingssummary %>%
  ggplot(aes(Pace,General_Position, color=Club_NameS))  + geom_point(size=3, show.legend=FALSE) +
  geom_text(aes(Pace,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, hjust =
```



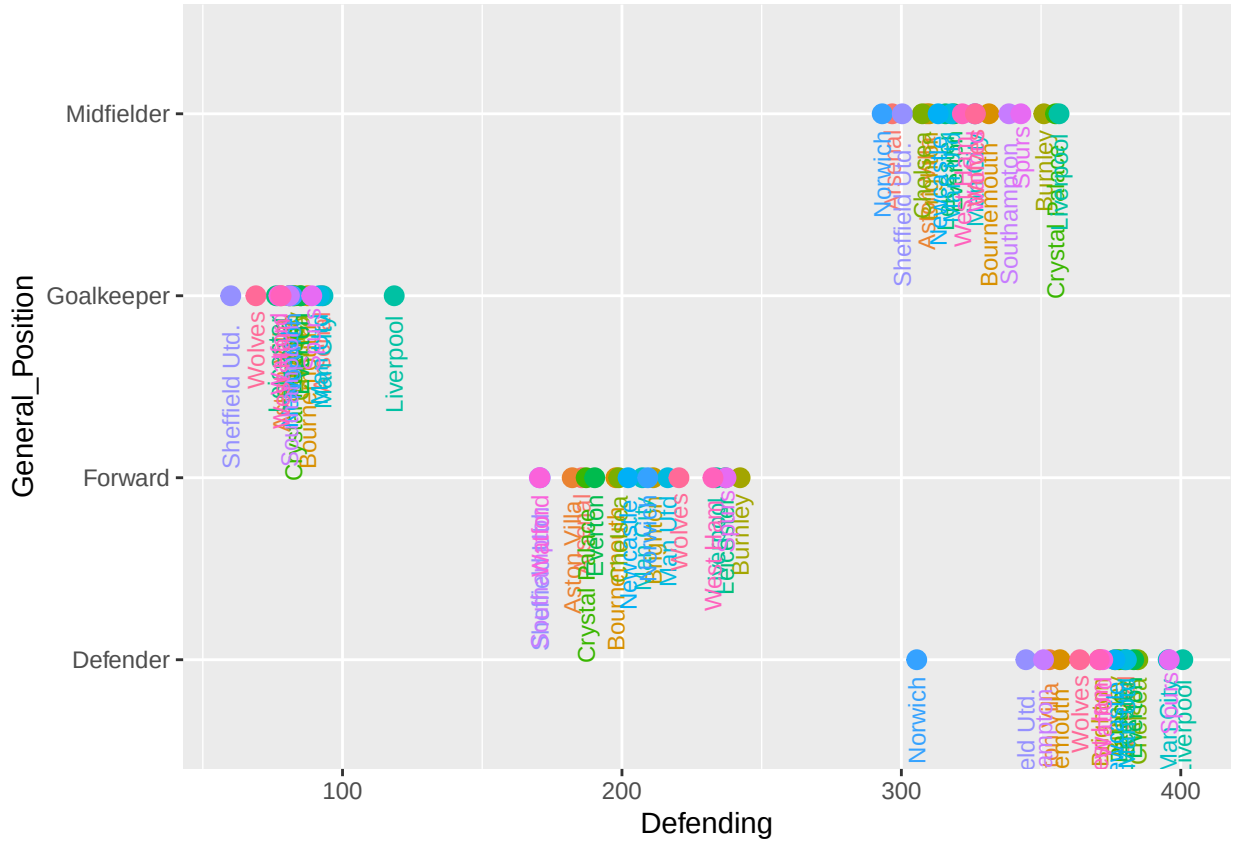

```
#shooting plot
Ratingssummary %>% ggplot(aes(Shooting,General_Position, color=Club_NameS))  + geom_point(size=3, show.
  geom_text(aes(Shooting,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, hjust
```



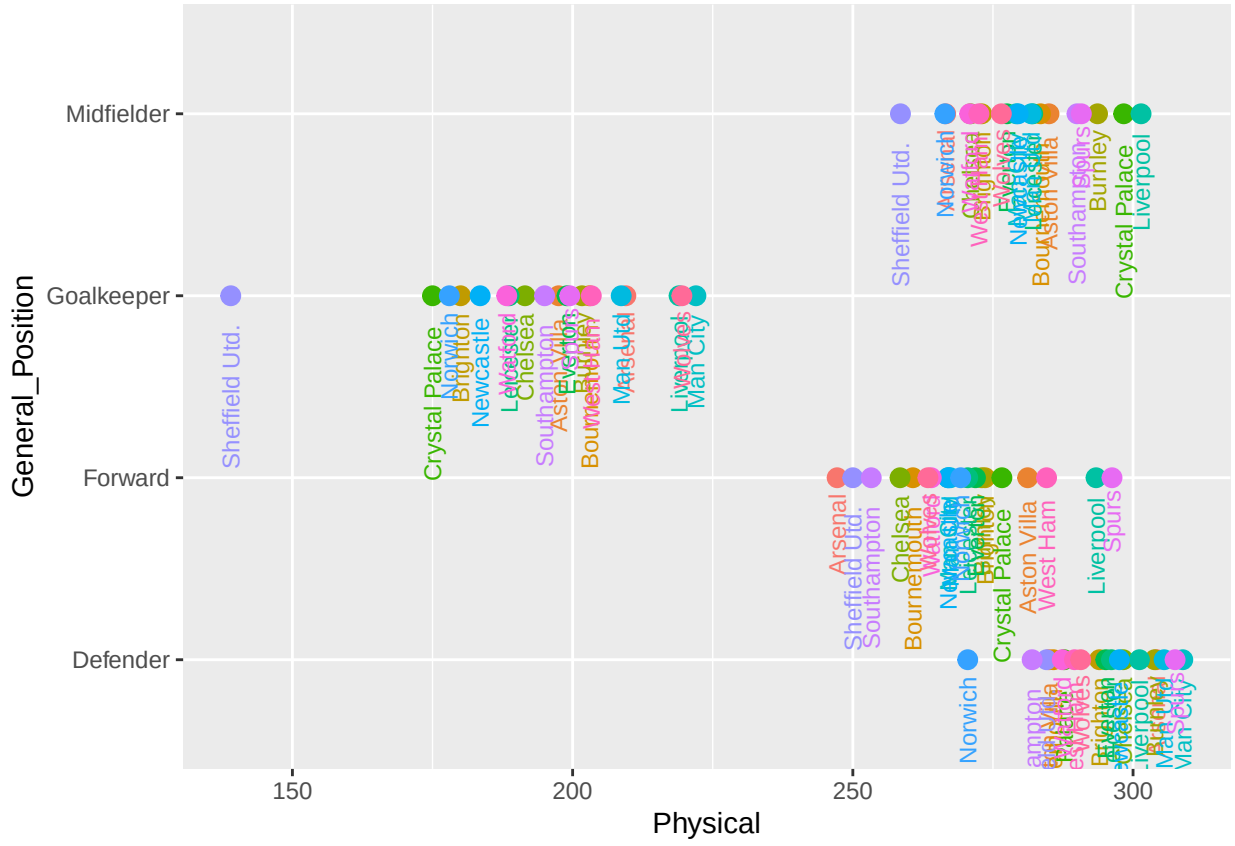

```
#Dribling plot
Ratingssummary %>% ggplot(aes(Dribbling,General_Position, color=Club_NameS))  + geom_point(size=3, show
  geom_text(aes(Dribbling,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, hju
```



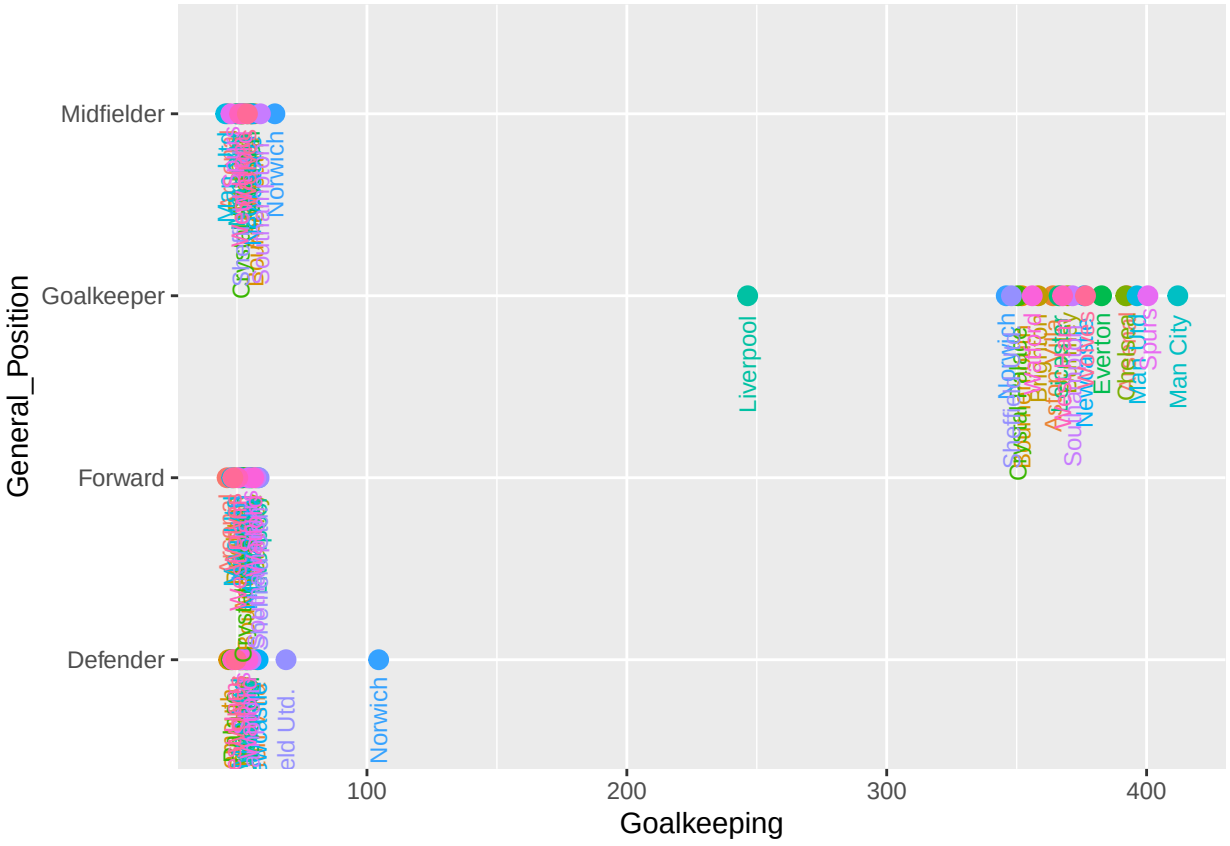

```
#Defending plot
Ratingssummary %>% ggplot(aes(Defending,General_Position, color=Club_NameS))  + geom_point(size=3, show
  geom_text(aes(Defending,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, hju
```




```
#Physical plot
Ratingssummary %>% ggplot(aes(Physical,General_Position, color=Club_NameS))  + geom_point(size=3, show.l
  geom_text(aes(Physical,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, hjust
```




```
#Goalkepping plot
Ratingssummary %>% ggplot(aes(Goalkeeping,General_Position, color=Club_NameS))  + geom_point(size=3, sh
  geom_text(aes(Goalkeeping,General_Position, label=Club_NameS),show.legend=FALSE, size= 3, angle=90, h
```



2.3 Datasets wrangling process for match results prediction model

2.3.1 organizing the initial data sets

To properly produce the models, the information included in both datasets need to be selected and organized. the initial step in this process is the reduction of the results dataset that include results from 1994 to only utilize the data from the season of 2019-2020 which is the FIFA ratings I want to include in the model.

```
S2019_20 <- EPL_results %>% filter(Season == "2019-20")
```

The FIFA ratings data base was organized to include first the character columns followed by the numeric columns

```
char<- EPL_fifa_ratings %>% select_if(is.character)%>% colnames() # selecting all the character columns
numer<- EPL_fifa_ratings %>% select_if(is.numeric)%>% colnames() #select all the numeric columns
EPL_fifa_ratings_r <- EPL_fifa_ratings[,c(char, numer)] # organize the columns locating the character c
```

As the results data set includes information for home team and away team it is necessary to include the ratings for the players of both teams in a singular dataset, the next sections describe how to create the home and away dataset and the complete data set.

2.3.2 Organizing HomeTeam data set

As I will join the two data sets using the club name, the first step was to rename the Club_NameS column in the ratings data set for HomeTeam which is the title in the results data set and test if the club names are

the same in both data sets

```
# change the column Club_NameS in ratings to have the same as season 2019-20
HomeTeam_ratings <- EPL_fifa_ratings_r %>% rename(HomeTeam=Club_NameS)

#test if all the club names are the same
identical(levels(as.factor(HomeTeam_ratings$HomeTeam)),levels(as.factor(S2019_20$HomeTeam)))
```

```
## [1] FALSE
```

As the club names were different in the data sets I evaluated which were the different names in the data set

```
A <- levels(as.factor(HomeTeam_ratings$HomeTeam)) # levels in ratings HomeTeam
B <- levels(as.factor(S2019_20$HomeTeam)) # levels in results HomeTeam

#Identify the club names missing in A
A[!(A%in%B)]
```

```
## [1] "Man Utd"        "Sheffield Utd." "Spurs"
```

with the Club names identified, I changed the names in the data sets and evaluate if the names were the same in both data sets.

```
#Homogenize the club names in the FIFA ratings and 2019-20 results
HomeTeam_ratings <- HomeTeam_ratings %>%
  mutate(HomeTeam = str_replace_all(HomeTeam_ratings$HomeTeam, "Man Utd", "Man United")) # replace club

HomeTeam_ratings_N <- HomeTeam_ratings %>%
  mutate(HomeTeam = str_replace_all(HomeTeam_ratings$HomeTeam, "Spurs", "Tottenham" )) # replace club n

S2019_20_N <- S2019_20 %>%
  mutate(HomeTeam = str_replace_all(S2019_20$HomeTeam, "Sheffield United","Sheffield Utd.")) #replace c

identical(levels(as.factor(HomeTeam_ratings_N$HomeTeam)),levels(as.factor(S2019_20_N$HomeTeam))) # chec
```

```
## [1] TRUE
```

As in the next steps the correlation matrix will be used to analyse the numeric variables I organized the data set to include the character variables followed by the numeric variables

```
# organize the datasets to have the characters columns first and the numeric columns later
S2019_20_N2 <- S2019_20_N[,c(1:4, 11, 7, 10, 5:6, 8:9, 12:23)]
```

As the levels in the column (HomeTeam) used for joining the two data sets are the same, I lef joined the two data sets using the next code

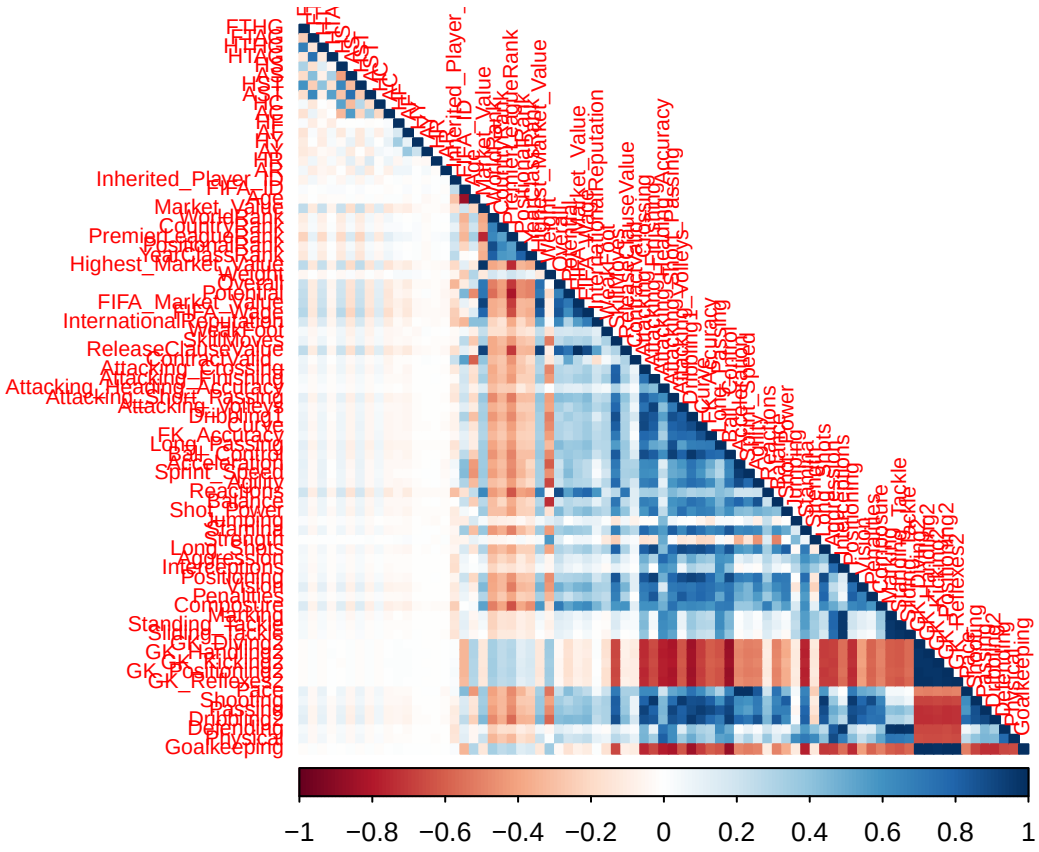
```
# Integration of HomeTeam's player's rating to results
Resuls_HomeRatings <- left_join(S2019_20_N2, HomeTeam_ratings_N, by= "HomeTeam" )
```

Similar to previous column organizations the data set was reorganized to include first the character columns and later the numeric columns, after that I added the summarizing variables I included earlier in the document.

```
#columns reorganization to facilitate further calculations
RH_num <- Resuls_HomeRatings %>% select_if(is.numeric)%>% colnames() #select all the numeric columns
RH_char <- Resuls_HomeRatings %>% select_if(is.character)%>% colnames() # select all the character colu
Resuls_HomeRatings_c <- Resuls_HomeRatings[,c(RH_char, RH_num )]  #organize the columns locating the ch

#add summarizing variables
Resuls_HomeRatings_c <-Resuls_HomeRatings_c %>%
  mutate(Pace=Acceleration+Sprint_Speed,
         Shooting= Attacking_Finishing+Long_Shots+Penalities+Positioning+Shot_Power+Attacking_Volleys,
         Passing=Attacking_Crossing + Curve + FK_Accuracy + Long_Passing + Attacking_Short_Passing + Vis
         Dribbling2= Agility + Balance + Ball_Control + Composure + Dribbling1 + Reactions,
         Defending= Attacking_Heading_Accuracy + Interceptions + Marking + Sliding_Tackle + Standing_Ta
         Physical= Aggression + Jumping + Stamina + Strength,
         Goalkeeping= GK_Diving2 + GK_Handling2 + GK_Kicking2 + GK_Positioning2 + GK_Reflexes2)
```

The correlation matrix for the home team data base was calculated and plotted using the following code and the numeric variables

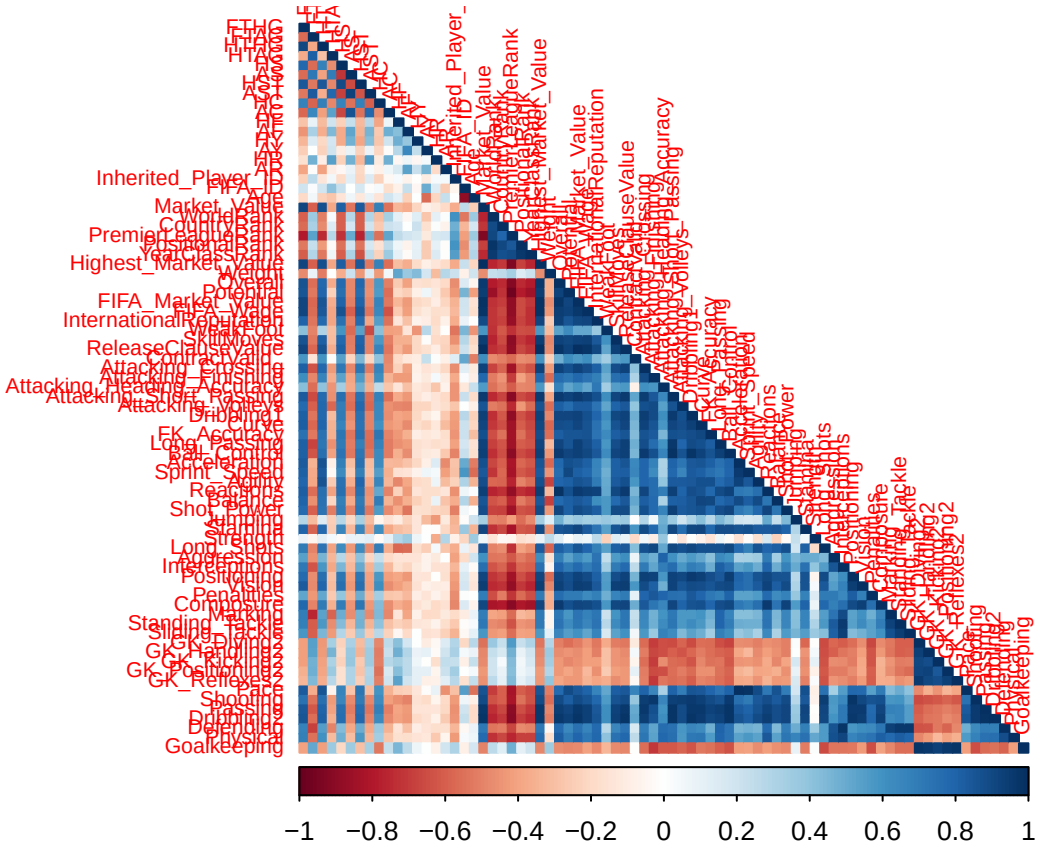
```
#correlation matrix using player ratings
Cor_play<- cor(Resuls_HomeRatings_c[28:104])


#correlation matrix plot
corrplot(Cor_play, method="color", type = "lower", tl.cex = 0.7 )
```



As the correlation between the FIFA ratings and the full time home team goals was low, I evaluated the

team ratings by averaging the ratings for each home team. With this summarized information, I recalculated the correlation matrix and the correlation plot

```
# summary of FIFA ratings by HomeTeam
HomeClub_summary <- Resuls_HomeRatings_c %>% group_by(HomeTeam)%>%
  summarise_at(vars(27:103), mean) %>% data.frame()

#calculation of club ratings correlation matrix
HTR<- cor(HomeClub_summary[2:78])


#Correlation plot using home team ratings
corrplot(HTR, method="color", type = "lower", tl.cex = 0.7 )
```



2.3.3 Organizing awayTeam data set

As I will join the two data sets using the club name, the first step was to rename the Club_NameS column in the ratings data set for AwayTeam which is the title in the results data set and test if the club names are the same in both data sets

```
# change the column Club.Name..short in ratings to have the same as season 2019-20
AwayTeam_ratings <- EPL_fifa_ratings_r %>% rename(AwayTeam=Club_NameS)

#test if all the club names are the same
identical(levels(as.factor(AwayTeam_ratings$AwayTeam)),levels(as.factor(S2019_20$AwayTeam)))
```

```
## [1] FALSE
```

As the club names were different in the data sets, I evaluated which were the different names in the data set

```
#As the names of the clubs were not identical we need to homogenize the club names in both datasets
x <- levels(as.factor(AwayTeam_ratings$AwayTeam))
y <- levels(as.factor(S2019_20$AwayTeam))

#Identify the club names missing in A
x[!(x%in%y)]
```

```
## [1] "Man Utd"        "Sheffield Utd." "Spurs"
```

with the Club names identified, I changed the names in the data sets and evaluate if the names were the same in both data sets.

```
#Homogenize the club names in the FIFA ratings and 2019-20 results
AwayTeam_ratings <- AwayTeam_ratings %>% mutate(AwayTeam = str_replace_all(AwayTeam_ratings$AwayTeam, "
AwayTeam_ratings_N <- AwayTeam_ratings %>% mutate(AwayTeam = str_replace_all(AwayTeam_ratings$AwayTeam,

S2019_20_AT <- S2019_20 %>% mutate(AwayTeam = str_replace_all(S2019_20$AwayTeam, "Sheffield United","Sh

identical(levels(as.factor(AwayTeam_ratings_N$AwayTeam)),levels(as.factor(S2019_20_AT$AwayTeam)))
```

```
## [1] TRUE
```

As the levels in the column (AwayTeam) used for joining the two datasets are the same, I left joined the two data sets using the next code.

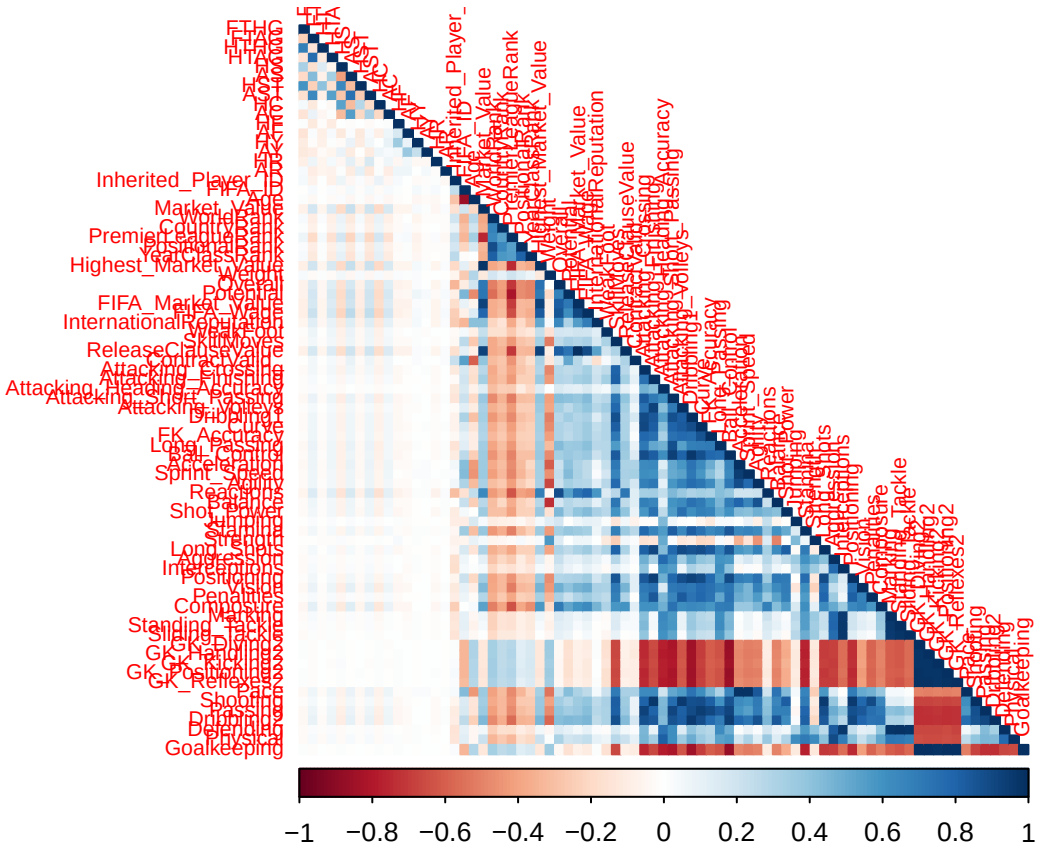
```
# Integration of AwayTeam's player's rating to results
Results_AwayRatings <- left_join(S2019_20_AT, AwayTeam_ratings_N , by= "AwayTeam" )
```

Similar to previous column organizations the data set was reorganized to include first the character columns and later the numeric columns, after that I added the summarizing variables I included earlier in the document.

```
#columns reorganization to facilitate further calculations
RA_num <- Results_AwayRatings  %>% select_if(is.numeric)%>% colnames() #select all the numeric columns
RA_char <- Results_AwayRatings  %>% select_if(is.character)%>% colnames() # select all the character co
Results_AwayRatings_c <- Results_AwayRatings [,c(RA_char, RA_num )]  #organize the columns locating the

#add summarizing variables
Results_AwayRatings_c <- Results_AwayRatings_c %>%
  mutate(Pace=Acceleration+Sprint_Speed,
         Shooting= Attacking_Finishing+Long_Shots+Penalities+Positioning+Shot_Power+Attacking_Volleys,
         Passing=Attacking_Crossing + Curve + FK_Accuracy + Long_Passing + Attacking_Short_Passing + Vi
         Dribbling2= Agility + Balance + Ball_Control + Composure + Dribbling1 + Reactions,
         Defending= Attacking_Heading_Accuracy + Interceptions + Marking + Sliding_Tackle + Standing_Ta
         Physical= Aggression + Jumping + Stamina + Strength,
         Goalkeeping= GK_Diving2 + GK_Handling2 + GK_Kicking2 + GK_Positioning2 + GK_Reflexes2)
```

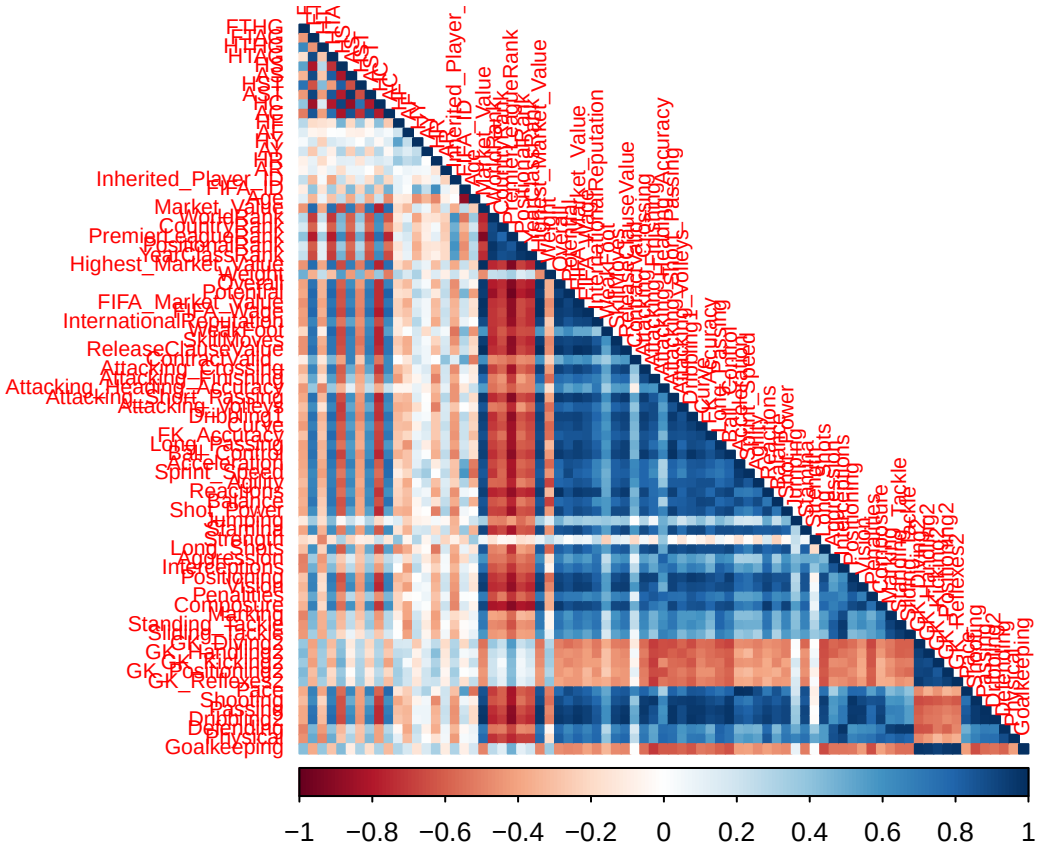
The correlation matrix for the home team data base was calculated and plotted using the following code and the numeric variables

```
#correlation matrix using player ratings
Cor_play_Away<- cor(Results_AwayRatings_c[28:104])

#correlation matrix plot
corrplot(Cor_play_Away, method="color", type = "lower", tl.cex = 0.7 )
```



As the correlation between the FIFA ratings and the full time away team goals is low, I will explore the team ratings by averaging the ratings for each away team. With this summarized information, I recalculated the correlation matrix and the correlation plot

```
# summary of fifa ratings by AwayTeam
Club_Away_summary <- Results_AwayRatings_c %>% group_by(AwayTeam)%>%
  summarise_at(vars(27:103), mean) %>% data.frame()

#calculation of correlation matrix
ATR<- cor(Club_Away_summary [2:78])

#correlation plot using Away team ratings
corrplot(ATR, method="color", type = "lower", tl.cex = 0.7 )
```



2.3.4 Combining the home and away team summarising data set

with the summarized version of the ratings, the home and away ratings for each team playing each match is included in the same data base

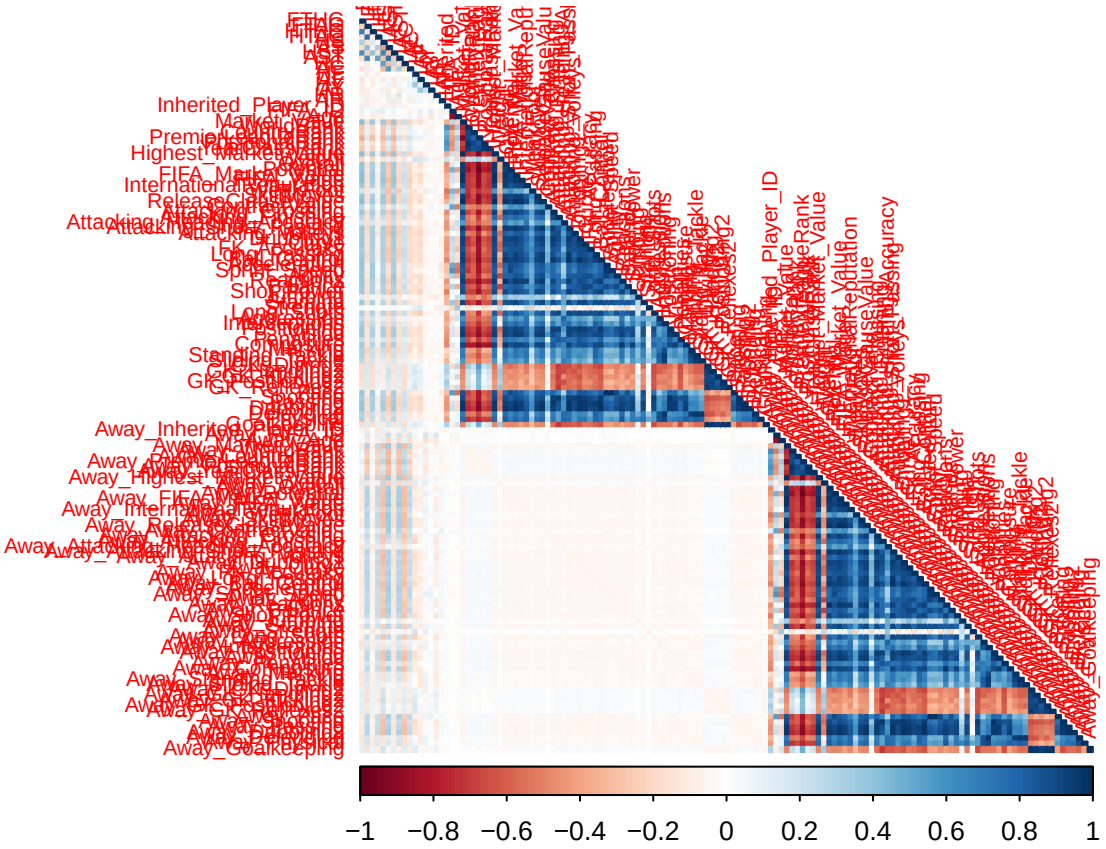
```
#integration of results with summarized datasets
#integration of results with summarised datasets
HomeClub_summary <- HomeClub_summary %>% select(1, 18:78) #remove the summarized results from homeclub
Club_Away_summary <- Club_Away_summary %>% select(1, 18:78) #remove the summarized results from awayclu

results_summary_club_home <- left_join(S2019_20_N, HomeClub_summary, "HomeTeam" ) # hometeam joined dat
RSA_num <- results_summary_club_home  %>% select_if(is.numeric)%>% colnames() #select all the numeric c
RSA_char <- results_summary_club_home  %>% select_if(is.character)%>% colnames() # select all the chara
results_summary_club_home <- results_summary_club_home [,c(RSA_char, RSA_num )]  #organize the columns

results_summary_club_Away <- left_join(S2019_20_AT, Club_Away_summary, "AwayTeam") #awayteam joined dat
RSH_num <- results_summary_club_Away %>% select_if(is.numeric)%>% colnames() #select all the numeric co
RSH_char <- results_summary_club_Away  %>% select_if(is.character)%>% colnames() # select all the chara
results_summary_club_home <- results_summary_club_home [,c(RSH_char, RSH_num )]  #organize the columns

#Whole data set integration
all<- cbind(results_summary_club_home, results_summary_club_Away[24:84])
```

to facilitate further calculations and avoid errors the columns were reorganized and the columns names from the away data set had an additional Away in the name

```
#renaming columns with duplicated names
colnames(all)[85:145]<-paste("Away", colnames(all)[85:145],sep="_" ) #add away to the away ratings colu

#columns reorganization to facilitate further calculations
all_num <- all %>% select_if(is.numeric)%>% colnames() #select all the numeric columns
all_char <- all  %>% select_if(is.character)%>% colnames() # select all the character columns
all <- all [,c(all_char, all_num )]  #organize the columns locating the character columns first followe
```

I tested the combined data set by calculating and plotting the correlation matrix

```
#calculation of correlation matrix
home_away_cor <- cor(all[8:145])

#correlation plot using home and away
corrplot(home_away_cor, method="color", type = "lower", tl.cex = 0.7 )
```



As both home team rating and away team ratings are correlated with the full time home goals and full time away goals.

I will evaluate different prediction model to predict the final results using the average FIFA team ratings with away teams, home teams and both teams.

3. RESULTS

3.1 TRAINING AND TEST SETS PARTITION

The initial step is the creation of training and test sets for the only home, only away and both teams data sets, using 20% of the data as the test.

```
# away ratings training and test data sets
set.seed(1, sample.kind = "Rounding")
test_index_away <- createDataPartition(results_summary_club_Away$FTR, times = 1, p = 0.2, list = FALSE)
test_set_away <- results_summary_club_Away[test_index_away,]
train_set_away <- results_summary_club_Away[-test_index_away,]

# Home ratings training and test data sets
set.seed(1, sample.kind = "Rounding")
test_index_home <- createDataPartition(results_summary_club_home$FTR, times = 1, p = 0.2, list = FALSE)
test_set_home <- results_summary_club_home[test_index_home,]
train_set_home <- results_summary_club_home[-test_index_home,]

# Home and away ratings training and test data sets
set.seed(1, sample.kind = "Rounding")
test_index_all <- createDataPartition(all$FTR, times = 1, p = 0.2, list = FALSE) # create a 20% test se
test_set_all <- all[test_index_all,]
train_set_all <- all[-test_index_all,]
```

3.2 GUESSING PREDICTION MODEL

The initial model tested is based in a guessing model using the FULL TIME RESULT (FTR) as the "Y" variable, this variable has 3 levels or results (Home team winning(H), Away team winning (A), Draw (D)). A montecarlo simulation was utilized to estimate the prediction accuracy after 10000 samples.

```
B <- 10^5
set.seed(1, sample.kind = "Rounding")
Away_guess <- replicate(B, {
  y_hat_guess_a <- sample(c("H", "A", "D"), length(test_index_away), replace = TRUE)
  mean(y_hat_guess_a == test_set_away$FTR)
  })

GA<- mean(Away_guess)  #mean accuracy of the simulation
GAmax <- max(Away_guess) #max accuracy obtained in the simulation
GAmin <- min(Away_guess) #min accuracy obtained in the simulation


# A model based on guessing the result using the Home data set and montecarlo simulation
B <- 10^5
set.seed(1, sample.kind = "Rounding")
Home_guess <- replicate(B, {
  y_hat_guess_h <- sample(c("H", "A", "D"), length(test_index_home), replace = TRUE)
  mean(y_hat_guess_h == test_set_home$FTR)
  })
GH<- mean(Home_guess)  # mean accuracy of the simulation
GHmax <- max(Home_guess) # max accuracy obtained in the simulation
GHmin <- min(Home_guess) #min accuracy obtained in the simulation

# A model based on guessing the result using the home and away data set and montecarlo simulation
```

```r
B <- 10^5
set.seed(1, sample.kind = "Rounding")
All_guess <- replicate(B, {
  y_hat_guess_all <- sample(c("H", "A", "D"), length(test_index_all), replace = TRUE)
  mean(y_hat_guess_all == test_set_all$FTR)
  })

GAll <- mean(All_guess) # mean accuracy of the simulation
GAllmax <- max(All_guess) # max accuracy obtained in the simulation
GAllmin <- min(All_guess)#min accuracy obtained in the simulation
```

```r
# a table including the accuracy of the three guess models from the three data sets
data.frame(Model=c("Away Team Ratings data", "Home Team Ratings data", "Home and Away Ratings data"),
          Mean_Accuracy=c(GA, GH,GAll), MaxAccuracy=c(GAmax, GHmax,GAllmax),
       MinAccuracy=c(GAmin, GHmin,GAllmin), MaxAccuracy_prob=c(mean(str_detect(Away_guess, "0.5641026"))
                                                               mean(str_detect(Home_guess, "0.5641026"))
                                                               mean(str_detect(All_guess, "0.5641026"))),
       MinAccuracy_prob=c(mean(str_detect(Away_guess, "0.1153846")),
                          mean(str_detect(Home_guess, "0.1153846")),
                          mean(str_detect(All_guess, "0.1153846"))))
```

Table 2. Accuracy for the different guessing models

```
##                         Model Mean_Accuracy MaxAccuracy MinAccuracy
## 1     Away Team Ratings data     0.3334287   0.5641026   0.1153846
## 2     Home Team Ratings data     0.3334287   0.5641026   0.1153846
## 3 Home and Away Ratings data     0.3334287   0.5641026   0.1153846
##   MaxAccuracy_prob MinAccuracy_prob
## 1          0.00023            1e-05
## 2          0.00023            1e-05
## 3          0.00023            1e-05
```

it was possible to observe that the guessing model prediction accuracy was close to the original probabilities of each result.

3.3 LDA PREDICTION MODEL

to improve the prediction accuracy I developed a LDA prediction model for the three data sets.

```r
# Model using only the away ratings and lda
train_lda1 <- train(train_set_away[25:84], train_set_away$FTR, method = "lda", data = train_set_away )
y_hat_aw <- predict(train_lda1, test_set_away )
Away_lda_ac <- confusionMatrix(data = y_hat_aw, reference = as.factor(test_set_away$FTR))$overall["Accu
Away_lda_var <- data.frame(confusionMatrix(data = y_hat_aw, reference = as.factor(test_set_away$FTR))$by
  select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence )
Awayldasum <-tibble(Dataset= "LDA - Away Team Ratings data", Match_result=c("A","D","H"), Away_lda_var )

# Model using only the home ratings and lda
train_lda_hom <- train(train_set_home[25:84], train_set_home$FTR, data =train_set_home, method = "lda")
```

```
y_hat_hom <- predict(train_lda_hom, test_set_home )
Home_lda_ac <-confusionMatrix(data = y_hat_hom, reference = as.factor(test_set_home$FTR))$overall["Accu
Home_lda_var <-data.frame(confusionMatrix(data = y_hat_hom, reference = as.factor(test_set_home$FTR))$by
                          select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence )
Homeldasum <-tibble(Dataset= "LDA - Home Team Ratings data", Match_result=c("A","D","H"), Home_lda_var

#varImp(train_lda_hom)

# Model using only the home and away ratings and lda

train_lda_all <- train(train_set_all[43:145], train_set_all$FTR, data =train_set_all, method = "lda")
y_hat_all <- predict(train_lda_all, test_set_all )
All_lda_ac <-confusionMatrix(data = y_hat_all, reference = as.factor(test_set_all$FTR))$overall["Accura
All_lda_var <- data.frame(confusionMatrix(data = y_hat_all, reference = as.factor(test_set_all$FTR))$by
                          select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence )
Alldasum <-tibble(Dataset= "LDA - Home and Away Team Ratings data", Match_result=c("A","D","H"), All_lda

#varImp(train_lda_all)
```

```
#LDA models accuracy table

LDAAccu<- tibble(Model=c("LDA-Away Team Ratings data", "LDA-Home Team Ratings data", "LDA-Home and Away
       Accuracy= c(Away_lda_ac,Home_lda_ac,All_lda_ac))
LDAAccu %>% knitr::kable()
```

TAble 3. LDA models accuracy

Model	Accuracy
LDA-Away Team Ratings data	0.4230769
LDA-Home Team Ratings data	0.4487179
LDA-Home and Away Ratings data	0.4487179

The accuracy of the models was higher when using the home team ratings and the home and away rating, in all the cases, the models improved the prediction accuracy compared with the guessing model.

```
#model summarizing table
LDA_summary <- bind_rows(Awayldasum, Homeldasum, Alldasum)
LDA_summary %>% knitr::kable()
```

Table 4. LDA models Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
LDA - Away Team Ratings data	A	0.4583333	0.6851852	0.3928571	0.5717593	0.3076923

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
LDA - Away Team Ratings data	D	0.2105263	0.8983051	0.4000000	0.5544157	0.2435897
LDA - Away Team Ratings data	H	0.5142857	0.4883721	0.4500000	0.5013289	0.4487179
LDA - Home Team Ratings data	A	0.3750000	0.7407407	0.3913043	0.5578704	0.3076923
LDA - Home Team Ratings data	D	0.3684211	0.7288136	0.3043478	0.5486173	0.2435897
LDA - Home Team Ratings data	H	0.5428571	0.6976744	0.5937500	0.6202658	0.4487179
LDA - Home and Away Team Ratings data	A	0.5416667	0.6481481	0.4062500	0.5949074	0.3076923
LDA - Home and Away Team Ratings data	D	0.1578947	0.8474576	0.2500000	0.5026762	0.2435897
LDA - Home and Away Team Ratings data	H	0.5428571	0.6511628	0.5588235	0.5970100	0.4487179

the three different match results had a balanced accuracy between 0.5 and 0.62, and a higher specificity than sensitivity. Each data set produced better balanced accuracy depending of the result Away team win prediction was more accurate using the home and away rating dataset, while, draws were more accurate using the away data set and home team wins are more accurate using the home team ratings data set. the lowest sensitivity is observed in draws indicating a significant difficulty in the model to detect draws. however, the model is able to identify non draws results as the specificity is relatively high. It is interesting to see that the sensitive for away team wins and draws is the most significant fluctuation in the models.

As all the LDA models generated a warning associated with colinearity among the variables I evaluated the effect of removing the predictor variables that are highly correlated among them (correlation higher than 0.95) and test if it was an improvement in the accuracy and colinearity. first the away team ratings data.

3.4 LDA NON-COLLINEAR PREDICTION MODEL

3.4.1 Removal of collinearity among predictors

To remove the colinearity among predictors I wrote a code to detect the predictors with a correlation between 0.95 and 1, and remove those from the initial datases. The correlation plots for the new set of predictors are included below

```
#removing collinearity from the away data set
w <- which( ATR>0.95 & ATR<1, arr.ind = TRUE) # selection of correlation between 0.95 and 1 in the cova
H <- w %>% as.data.frame() %>% group_by(col)%>% summarise(n=n()) %>% arrange(col) %>% data.frame() # id
Ss<-H$col+ (ncol(results_summary_club_Away)-ncol(ATR))  # identification of the columns number in the r
Away_nocolinear_data <- results_summary_club_Away%>% select(!Ss) #removing colinear columns
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(Ss)‘ instead of ‘Ss‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
#Cor_play_Awaynocol<- cor(Away_nocolinear_data[8:ncol(Away_nocolinear_data)]) #correlation matrix
#corrplot(Cor_play_Awaynocol, method="color", type = "lower", tl.cex = 0.7 )  #correlation plot
```

```
#removing collinearity from the home data set
hn <- which( HTR>0.95 & HTR<1, arr.ind = TRUE) # selection of correlation between 0.95 and 1 in the cov
HN<- hn %>% as.data.frame() %>% group_by(col)%>% summarise(n=n()) %>% arrange(col) %>% tibble() # ident
Ssn<-HN$col+7 # identification of the columns number in the results data set using the information from
Home_nocolinear_data <- results_summary_club_home%>% select(!Ssn) #removing colinear columns
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(Ssn)‘ instead of ‘Ssn‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
#Cor_play_homenocol<- cor(Home_nocolinear_data[9:ncol(Home_nocolinear_data)]) #correlation matrix
#corrplot(Cor_play_homenocol, method="color", type = "lower", tl.cex = 0.7 ) #correlation plot

#removing colinearity from the home and away data set
an <- which( home_away_cor>0.95 & home_away_cor<1, arr.ind = TRUE) # selection of correlation between 0
aN<- an %>% as.data.frame() %>% group_by(col)%>% summarise(n=n()) %>% arrange(col) %>% tibble() # ident
Asn<-aN$col+7 # identification of the columns number in the results data set using the information from
all_nocolinear_data <- all%>% select(!Asn)#removing colinear columns
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(Asn)‘ instead of ‘Asn‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
#Cor_play_allnocol<- cor(all_nocolinear_data[9:ncol(all_nocolinear_data)]) #correlation matrix
#corrplot(Cor_play_allnocol, method="color", type = "lower", tl.cex = 0.7 )#correlation plot
```

3.4.2 LDA models using the no-colinear data

The new data sets including the noncolinear data produced different predictors for each data set

```
#evaluating the lda model using the noncolinear away data set
set.seed(1, sample.kind = "Rounding")
test_index_nocol <- createDataPartition(Away_nocolinear_data$FTR, times = 1, p = 0.2, list = FALSE) # c
test_set_away_ncol <- Away_nocolinear_data[test_index_nocol,] #non-collinear test set
train_set_away_ncol <- Away_nocolinear_data[-test_index_nocol,] # non-collinear train test
train_lda11 <- train(train_set_away_ncol[25:ncol(Away_nocolinear_data)], train_set_away_ncol$FTR,
                     method = "lda", data = train_set_away_ncol )
y_hat_aw1 <- predict(train_lda11, test_set_away_ncol  )
Awaynoc_lda_ac <-confusionMatrix(data = y_hat_aw1, reference = as.factor(test_set_away_ncol$FTR))$overal
Awaynoc_lda_var <-data.frame(confusionMatrix(data = y_hat_aw1, reference = as.factor(test_set_away_ncol
                             select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalenc
Awaynoc_ldasum <-tibble(Dataset= "LDA-No colinear Away Team Ratings data", Match_result=c("A","D","H"),


#evaluating the lda model using the noncolinear home data set
set.seed(1, sample.kind = "Rounding")
test_index_hnc <- createDataPartition(Home_nocolinear_data$FTR, times = 1, p = 0.2, list = FALSE) # cre
test_set_home_hnc <- Home_nocolinear_data[test_index_hnc,]
```

```
train_set_home_hnc <- Home_nocolinear_data[-test_index_hnc,]
train_lda2 <- train(train_set_home_hnc[25:ncol(Home_nocolinear_data)], train_set_home_hnc$FTR,
                    method = "lda", data = train_set_home_hnc )
y_hat_hom <- predict(train_lda2, test_set_home )
Homenoc_lda_ac <- confusionMatrix(data = y_hat_hom, reference = as.factor(test_set_home_hnc$FTR))$overal
Homenoc_lda_var <- data.frame(confusionMatrix(data = y_hat_hom, reference = as.factor(test_set_home_hnc$
                              select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalenc
Homenoc_ldasum <-tibble(Dataset= "LDA-No colinear Home Team Ratings data",
                        Match_result=c("A","D","H"), Homenoc_lda_var ) #model information table


#evaluating the lda model using the noncolinear home and away data set
set.seed(1, sample.kind = "Rounding")
test_index <- createDataPartition(all_nocolinear_data $FTR, times = 1, p = 0.2, list = FALSE) # create
test_set_all <- all_nocolinear_data [test_index,]
train_set_all <- all_nocolinear_data [-test_index,]
train_lda3 <- train(train_set_all[25:ncol(all_nocolinear_data)], train_set_all$FTR, method = "lda", data
y_hat_all <- predict(train_lda3, test_set_all )
Allnoc_lda_ac <- confusionMatrix(data = y_hat_all, reference = as.factor(test_set_all$FTR))$overall["Ac
Allnoc_lda_var <- data.frame(confusionMatrix(data = y_hat_all, reference = as.factor(test_set_all$FTR))$
                             select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalenc
Allnoc_ldasum <-tibble(Dataset= "LDA-No colinear Home and Away Team Ratings data",
                        Match_result=c("A","D","H"), Allnoc_lda_var ) #model information table
```

```
#no collinear LDA models accuracy table
LDAAccu<- tibble(Model=c("Non collinear LDA - Away Team Ratings data",
                         "Non collinear LDA - Home Team Ratings data",
                         "Non collinear LDA - Home and Away Ratings data"),
                 Accuracy= c(Awaynoc_lda_ac,Homenoc_lda_ac,Allnoc_lda_ac))
LDAAccu %>% knitr::kable()
```

TAble 5. non-collinear LDA models accuracy

Model	Accuracy
Non collinear LDA - Away Team Ratings data	0.4230769
Non collinear LDA - Home Team Ratings data	0.4487179
Non collinear LDA - Home and Away Ratings data	0.4487179

The accuracy of the colinear and noncolinear were almost the same. evidencing that the number of predictors did not have a strong influence in the model or that the colinearity had a significant effect in the LDA models.

```
#model summarizing table
LDA_summary <- bind_rows(Awaynoc_ldasum, Homenoc_ldasum, Allnoc_ldasum)
LDA_summary %>% knitr::kable()
```

Table 6. non-collinear LDA models Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
LDA-No colinear Away Team Ratings data	A	0.4583333	0.6851852	0.3928571	0.5717593	0.3076923
LDA-No colinear Away Team Ratings data	D	0.2105263	0.8983051	0.4000000	0.5544157	0.2435897
LDA-No colinear Away Team Ratings data	H	0.5142857	0.4883721	0.4500000	0.5013289	0.4487179
LDA-No colinear Home Team Ratings data	A	0.3750000	0.7407407	0.3913043	0.5578704	0.3076923
LDA-No colinear Home Team Ratings data	D	0.3684211	0.7288136	0.3043478	0.5486173	0.2435897
LDA-No colinear Home Team Ratings data	H	0.5428571	0.6976744	0.5937500	0.6202658	0.4487179
LDA-No colinear Home and Away Team Ratings data	A	0.5416667	0.6481481	0.4062500	0.5949074	0.3076923
LDA-No colinear Home and Away Team Ratings data	D	0.1578947	0.8474576	0.2500000	0.5026762	0.2435897
LDA-No colinear Home and Away Team Ratings data	H	0.5428571	0.6511628	0.5588235	0.5970100	0.4487179

Similar to accuracy, the sensitivity and specificity did not changed with the predictors and colinearity reduction.

3.5 KNN MODELS

```
# knn model with away data
set.seed(1, sample.kind = "Rounding") # simulate R 3.5
tuning <- data.frame(k = seq(3, 55, 1))
awaytrain_knn <- train(train_set_away[25:ncol(train_set_away)], train_set_away$FTR,
                       method = "knn", tuneGrid = tuning )
awayknn_preds <- predict(awaytrain_knn, test_set_away)
awaytrain_knn$bestTune
```

```
##     k
## 53 55
```

```
away_knn_ac <- confusionMatrix(data = awayknn_preds, reference = as.factor(test_set_away$FTR))$overall[
away_knn_var <-data.frame(confusionMatrix(data = awayknn_preds, reference = as.factor(test_set_away$FTR
  select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
away_knnsum <-tibble(Dataset= "KNN model - Away Team Ratings data",
                     Match_result=c("A","D","H"),away_knn_var ) #model information table

# knn model with home data
set.seed(1, sample.kind = "Rounding") # simulate R 3.5
tuning <- data.frame(k = seq(3, 55, 1))
hometrain_knn <- train(train_set_home[25:ncol(train_set_home)], train_set_home$FTR,
                       method = "knn", tuneGrid = tuning )
```

```
homeknn_preds <- predict(hometrain_knn, test_set_home)
hometrain_knn$bestTune
```

```
##     k
## 10 12
```

```
home_knn_ac <- confusionMatrix(data = homeknn_preds, reference = as.factor(test_set_home$FTR))$overall[
home_knn_var <-data.frame(confusionMatrix(data = homeknn_preds, reference = as.factor(test_set_home$FTR
  select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
home_knnsum <-tibble(Dataset= "KNN model - Home Team Ratings data",
                     Match_result=c("A","D","H"),home_knn_var ) #model information table

# knn model with home and away data
set.seed(1, sample.kind = "Rounding") # simulate R 3.5
tuning <- data.frame(k = seq(3, 55, 1))
train_knn <- train(train_set_all[25:ncol(train_set_all)], train_set_all$FTR,
                   method = "knn", tuneGrid = tuning )
knn_preds <- predict(train_knn, test_set_all)
train_knn$bestTune
```

```
##     k
## 10 12
```

```
All_knn_ac <- confusionMatrix(data = knn_preds, reference = as.factor(test_set_all$FTR))$overall["Accura
All_knn_var <-data.frame(confusionMatrix(data = knn_preds, reference = as.factor(test_set_all$FTR))$byC
                           select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
All_knnsum <-tibble(Dataset= "KNN model - Home and Away Team Ratings data",
                    Match_result=c("A","D","H"), All_knn_var ) #model information table
```

the best k for the away team ratings dataset model was 55, while for the home team rating dataset and the combined home and away teams rating dataset, the best k was 12.

```
 # knn  models accuracy table
knn_Accu<- tibble(Model=c("knn - Away Team Ratings data",
                          "knn - Home Team Ratings data",
                          "knn - Home and Away Ratings data"),
                  Accuracy= c(away_knn_ac,home_knn_ac,All_knn_ac))
knn_Accu %>% knitr::kable()
```

Table 7. knn models accuracy

Model	Accuracy
knn - Away Team Ratings data	0.4230769
knn - Home Team Ratings data	0.4743590
knn - Home and Away Ratings data	0.5000000

knn models obtained better accuracy in the home team ratings and the combined home and away teams

ratings than previous models. The knn models of the Home team ratings data improved 5% compared to the LDA models, while the combined home and away teams ratings dataset improved the accuracy in 10% versus the LDA models and 33% versus the guessing model. The away data ratings dataset did not improve its accuracy.

```
# knn models summarizing table
knn_summary <- bind_rows(away_knnsum , home_knnsum, All_knnsum )
knn_summary %>% knitr::kable()
```

Table 8. knn models Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
KNN model - Away Team Ratings data	A	0.2500000	0.7777778	0.3333333	0.5138889	0.3076923
KNN model - Away Team Ratings data	D	0.0000000	1.0000000	NA	0.5000000	0.2435897
KNN model - Away Team Ratings data	H	0.7714286	0.2325581	0.4500000	0.5019934	0.4487179
KNN model - Home Team Ratings data	A	0.3750000	0.7222222	0.3750000	0.5486111	0.3076923
KNN model - Home Team Ratings data	D	0.1578947	0.8983051	0.3333333	0.5280999	0.2435897
KNN model - Home Team Ratings data	H	0.7142857	0.5348837	0.5555556	0.6245847	0.4487179
KNN model - Home and Away Team Ratings data	A	0.5000000	0.8148148	0.5454545	0.6574074	0.3076923
KNN model - Home and Away Team Ratings data	D	0.1052632	0.8983051	0.2500000	0.5017841	0.2435897
KNN model - Home and Away Team Ratings data	H	0.7142857	0.4651163	0.5208333	0.5897010	0.4487179

The knn models had significant sensitivity towards home team wins as the three data sets reached a sensitivity above 0.7. Similar to lda, the combined home and away data set had the highest scores for balanced accuracy and sensitivity. the significant improvement in accuracy is associate with the significant improvement predicting home team wins.

3.6 RANDOM FOREST MODELS

```
#model using away team ratings data set and random forest
control <- trainControl(method="cv", number = 5, p = 0.8)
grid <- expand.grid(minNode = c(1:15) , predFixed = c(10, 15, 25, 35, 50))
awaytrain_rf <-  train(train_set_away[25:ncol(train_set_away)], train_set_home$FTR,
                       method = "Rborist",
                       nTree = 50,
                       trControl = control,
                       tuneGrid = grid,
                       nSamp = 5000) #model training
```

```
awaytrain_rf$bestTune$minNode #BEST MIN NODE
```

```
## [1] 4
```

```
awaytrain_rf$bestTune$predFixed # BEST PREDFIXED
```

```
## [1] 50
```

```
awaypred_rf <- Rborist(train_set_away[25:ncol(train_set_away)], as.factor(train_set_away$FTR),
                       nTree = 1000,
                       minNode = awaytrain_rf$bestTune$minNode,
                       predFixed = awaytrain_rf$bestTune$predFixed) #model prediction

y_hat_awayrf <- predict(awaypred_rf, test_set_away[25:ncol(test_set_away)])$yPred
away_RF_ac <-confusionMatrix(data = y_hat_awayrf, reference = as.factor(test_set_away$FTR))$overall["Ac
away_RF_var <- data.frame(confusionMatrix(data = y_hat_awayrf, reference = as.factor(test_set_away$FTR)
  select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
away_RFsum <-tibble(Dataset= "RF model - Away Team Ratings data",
                    Match_result=c("A","D","H"), away_RF_var ) #model information table



#model using home team ratings data set and random forest
control <- trainControl(method="cv", number = 5, p = 0.8)
grid <- expand.grid(minNode = c(1,15) , predFixed = c(10, 15, 25, 35, 50))
hometrain_rf <-  train(train_set_home[25:ncol(train_set_home)], train_set_home$FTR,
                   method = "Rborist",
                   nTree = 50,
                   trControl = control,
                   tuneGrid = grid,
                   nSamp = 5000) #model training

hometrain_rf$bestTune$minNode #BEST MIN NODE
```

```
## [1] 1
```

```
hometrain_rf$bestTune$predFixed # BEST PREDFIXED
```

```
## [1] 50
```

```
homepred_rf <- Rborist(train_set_home[25:ncol(train_set_home)], as.factor(train_set_home$FTR),
                   nTree = 1000,
                   minNode = hometrain_rf$bestTune$minNode,
                   predFixed = hometrain_rf$bestTune$predFixed) #model prediction

y_hat_homerf <- predict(homepred_rf, test_set_home[25:ncol(test_set_home)])$yPred
home_RF_ac <-confusionMatrix(data = y_hat_homerf, reference = as.factor(test_set_home$FTR))$overall["Ac
home_RF_var <- data.frame(confusionMatrix(data = y_hat_homerf, reference = as.factor(test_set_home$FTR)
  select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
home_RFsum <-tibble(Dataset= "RF model - Home Team Ratings data",
```

```
                  Match_result=c("A","D","H"), home_RF_var ) #model information table


#model using home and away ratings data set and random forest
control <- trainControl(method="cv", number = 5, p = 0.8)
grid <- expand.grid(minNode = c(1,15) , predFixed = c(10, 15, 25, 35, 50))
train_rf <-  train(train_set_all[25:ncol(train_set_all)], train_set_all$FTR,
                   method = "Rborist",
                   nTree = 50,
                   trControl = control,
                   tuneGrid = grid,
                   nSamp = 5000) #model training

train_rf$bestTune$minNode #BEST MIN NODE
```

```
## [1] 1
```

```
train_rf$bestTune$predFixed # BEST PREDFIXED
```

```
## [1] 10
```

```
pred_rf <- Rborist(train_set_all[25:ncol(train_set_all)], as.factor(train_set_all$FTR),
                   nTree = 1000,
                   minNode = train_rf$bestTune$minNode,
                   predFixed = train_rf$bestTune$predFixed) #model prediction

y_hat_rf <- predict(pred_rf, test_set_all[25:ncol(train_set_all)])$yPred
All_RF_ac <-confusionMatrix(data = y_hat_rf, reference = as.factor(test_set_all$FTR))$overall["Accuracy
All_RF_var <- data.frame(confusionMatrix(data = y_hat_rf, reference = as.factor(test_set_all$FTR))$byCl
                         select(Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence)
All_RFsum <-tibble(Dataset= "RF model - Home and Away Team Ratings data",
                    Match_result=c("A","D","H"), All_RF_var ) #model information table



 # knn  models accuracy table
knn_Accu<- tibble(Model=c("knn - Away Team Ratings data",
                          "knn - Home Team Ratings data",
                          "knn - Home and Away Ratings data"),
                  Accuracy= c(away_RF_ac,home_RF_ac,All_RF_ac))
knn_Accu %>% knitr::kable()
```

Table 9. Random forest models accuracy

Model	Accuracy
knn - Away Team Ratings data	0.4358974
knn - Home Team Ratings data	0.4615385
knn - Home and Away Ratings data	0.4230769

The random forest's accuracy was lower than the knn models, however, it is interesting that the home team ratings data had the highest accuracy when for the other models the combined home and away data set presented the better results.

```
# knn models summarizing table
knn_summary <- bind_rows(away_RFsum , home_RFsum, All_RFsum )
knn_summary %>% knitr::kable()
```

Table 10. Random forest models Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
RF model - Away Team Ratings data	A	0.5416667	0.6296296	0.3939394	0.5856481	0.3076923
RF model - Away Team Ratings data	D	0.0526316	0.9491525	0.2500000	0.5008921	0.2435897
RF model - Away Team Ratings data	H	0.5714286	0.5116279	0.4878049	0.5415282	0.4487179
RF model - Home Team Ratings data	A	0.3750000	0.7407407	0.3913043	0.5578704	0.3076923
RF model - Home Team Ratings data	D	0.3684211	0.7796610	0.3500000	0.5740410	0.2435897
RF model - Home Team Ratings data	H	0.5714286	0.6511628	0.5714286	0.6112957	0.4487179
RF model - Home and Away Team Ratings data	A	0.5416667	0.6851852	0.4333333	0.6134259	0.3076923
RF model - Home and Away Team Ratings data	D	0.1578947	0.7796610	0.1875000	0.4687779	0.2435897
RF model - Home and Away Team Ratings data	H	0.4857143	0.6511628	0.5312500	0.5684385	0.4487179

Although, the random forest model did not achieve the highest accuracy, it did achieve the highest balanced accuracy and sensitivity of any model to detect draws. the highest accuracy obtained by the home team ratings data set is associated with the greater sensitivity with draws, however, the same model had a lower sensitivity towards away team wins.

3.7 MODEL SELECTION

As the data set combining the home and away teams ratings performed the best, I compared the 4 methods used with that data set.

```
#home and away accuracy
whole_Accu<- tibble(Model=c( "Guess Model - Home and Away Ratings data",
                             "LDA-Home and Away Ratings data",
                             "Non collinear LDA - Home and Away Ratings data",
                             "knn - Home and Away Ratings data",
```

```
                              "RF - Home and Away Ratings data"),
                    Accuracy= c(GAll,All_lda_ac, Allnoc_lda_ac,All_knn_ac,All_RF_ac))
whole_Accu %>% knitr::kable()
```

Table 11. Random forest models accuracy

Model	Accuracy
Guess Model - Home and Away Ratings data	0.3334287
LDA-Home and Away Ratings data	0.4487179
Non collinear LDA - Home and Away Ratings data	0.4487179
knn - Home and Away Ratings data	0.5000000
RF - Home and Away Ratings data	0.4230769

As exposed before the best method to predict the EPL matches was knn, as it makes a good prediction 50% of the times. although the prediction is not perfect is a good initial approach to use the players' ratings from fifa videogames as predicting variables for match results.

```
# random forest models summarizing table
whole_summary <- bind_rows( Alldasum, Allnoc_ldasum, All_knnsum, All_RFsum )
whole_summary %>% knitr::kable()
```

Table 12. models comparison Sensitivity, Specificity, Precision, Balanced.Accuracy, Prevalence

Dataset	Match_result	Sensitivity	Specificity	Precision	Balanced.Accuracy	Prevalence
LDA - Home and Away Team Ratings data	A	0.5416667	0.6481481	0.4062500	0.5949074	0.3076923
LDA - Home and Away Team Ratings data	D	0.1578947	0.8474576	0.2500000	0.5026762	0.2435897
LDA - Home and Away Team Ratings data	H	0.5428571	0.6511628	0.5588235	0.5970100	0.4487179
LDA-No colinear Home and Away Team Ratings data	A	0.5416667	0.6481481	0.4062500	0.5949074	0.3076923
LDA-No colinear Home and Away Team Ratings data	D	0.1578947	0.8474576	0.2500000	0.5026762	0.2435897
LDA-No colinear Home and Away Team Ratings data	H	0.5428571	0.6511628	0.5588235	0.5970100	0.4487179
KNN model - Home and Away Team Ratings data	A	0.5000000	0.8148148	0.5454545	0.6574074	0.3076923
KNN model - Home and Away Team Ratings data	D	0.1052632	0.8983051	0.2500000	0.5017841	0.2435897
KNN model - Home and Away Team Ratings data	H	0.7142857	0.4651163	0.5208333	0.5897010	0.4487179
RF model - Home and Away Team Ratings data	A	0.5416667	0.6851852	0.4333333	0.6134259	0.3076923
RF model - Home and Away Team Ratings data	D	0.1578947	0.7796610	0.1875000	0.4687779	0.2435897
RF model - Home and Away Team Ratings data	H	0.4857143	0.6511628	0.5312500	0.5684385	0.4487179

he knn model obtained a higher accuracy compared with the other models, this higher accuracy is associated with a higher sensitivity toward the home team wins. however, the specificity of this prediction is still low and can be improved. Although, the knn model has higher accuracy, it has less sensitivity predicting away team wins and draws than the other models. The combined data set evidenced better results especially for the draws and the away teams. however, the predictors from the fifa videogame were not enough to achieve sensitivities above 55% for these two results.

In the future, it is necessary to add other predictors that can be associated with draws or away wins. Therefore, the sensitivity and specificity of the prediction can improve. Additionally, it will be interesting if it is possible to include formations, injuries and other variables from other data sets. The fifa videogame players' rating can be included with other variables to improve match results prediction.

4. CONCLUSION

The player's ratings from the FIFA 2020 videogame were able to predict match results from the English premier league with accuracy levels between 0.45 and 0.5. The three player's rating datasets were useful to do predictions, however, the combined home and away data set produced the best accuracy. knn model was selected has the better option as it obtained the highest overall accuracy and the highest sensitivity towards home team wins. In the future, the FIFA player's ratings dataset can be combined with other types of information associated with the home and away teams and with variables that can give more insight about the draws or the away team wins.