Analysis on English Premier League Teams

Purpose of the study

The English Premier League is one of the most watched sports leagues in the world. Implications of winning a championship is extremely important and as a result, statistics and data are a large component of a modern day premier league club. The main purpose of my study is twofold: (1) To determine which metric is the best indicator of a team's success. (2) To predict which team will win the premier league according to those metrics.

Data

There are a multitude of websites that provide extensive premier league data, but I decided to settle on fbref, which according to the wider community, is reliable. The next step was to extract data for 6 consecutive seasons, starting with the 2018/19 season, until the current season, 2023/24. I chose to focus on the following tables: Regular season, Squad Standard Stats, Squad Goal and Shot Creation, and Squad Defensive Actions. These tables have the most impactful metrics that summarize a team's performance, and so are the most relevant to our study. To execute the extraction, I used python, using the pandas library to extract data from the specific season's url. I then downloaded the tables as csv files that I stored on my local computer.

The csv files were then opened on Rstudio, where I would complete the remainder of the project. After reading in the csv files, the next step in the process of preparing the data was to clean the data. In this step, I skipped over the first row for some files, and for the Regular Season data, extracted the specific columns relevant to my study (Squad, Wins, Draws, and Losses). Prior to merging, I need to add an identifier for each squad's statistics by season. So, I created a column for each table's data, which has the

season the statistics were measured in. Next, I had to combine all the data into one dataframe, which I did using merge(). Furthermore, I calculated the points tally for each team, multiplying the number of wins by three and the number of draws by one. Now, the data was ready for analysis.

Analysis, e.g., summary statistics, visualization, models

I first performed correlation analysis, calculating the correlation between each metric in the dataframe with the total number of points using cor().

```
  X.x    X..Pl.x        Age       Poss         MP     Starts        Min     X90s.x        Ast        G.A
[1,] -0.05515308 -0.3234438 -0.01254734 0.7709775 -0.01691894 -0.01691894 -0.01691894
-0.01691894 0.6838996 0.6895299
       G.PK        PK     PKatt      CrdY      CrdR        xG      npxG       xAG  npxG.xAG      PrgC      PrgP
[1,] 0.6786337 0.4403689 0.3978793 -0.2770981 -0.1834171 0.5813917 0.5721402 0.5817371
0.5778091 0.4650045 0.496815
       xG.1     xAG.1    xG.xAG    npxG.1 npxG.xAG.1         W         D         L         X.y    X..Pl.y
[1,] 0.8516836 0.8302355 0.8495447 0.8359993  0.8374943 0.7839217 -0.09749113 -0.7908339
-0.05515308 -0.3111508
       X90s.y       Tkl      TklW    Def.3rd    Mid.3rd   Att.3rd      Tkl.1       Att       Tkl.      Lost
[1,] -0.01287001 -0.1697672 -0.150056 -0.3355234 -0.05640638 0.2316476 -0.1716392 -0.1479088
-0.05584914 -0.1231502
       Blocks       Sh.x       Pass        Int    Tkl.Int        Clr        Err          X      X..Pl       X90s
[1,] -0.2712803 -0.4449459 -0.1398369 -0.2350329 -0.2031795 -0.3653986 0.01838646 -0.05515308
-0.3234438 -0.01691894
       SCA     SCA90  PassLive   PassDead         TO       Sh.y        Fld        Def        GCA     GCA90
PassLive.1
```

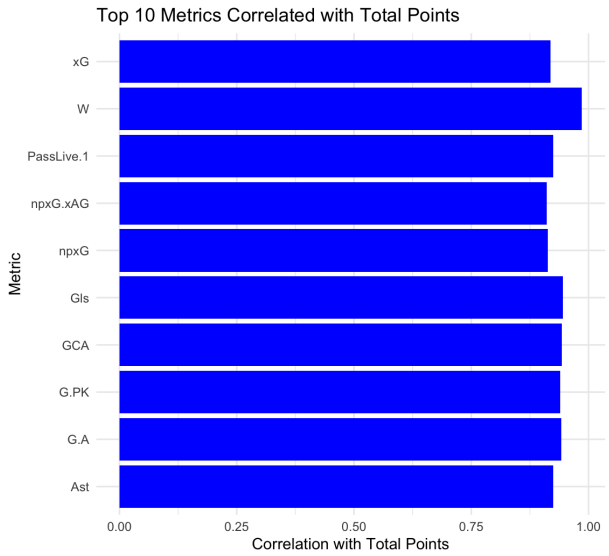
[1,] 0.443718 0.7974633 0.4863269 0.03928205 0.3934961 0.3951654 0.0539212 0.1057919 0.6978284 0.885953 0.7064955

PassDead.1 TO.1 Sh.1 Fld.1 Def.1

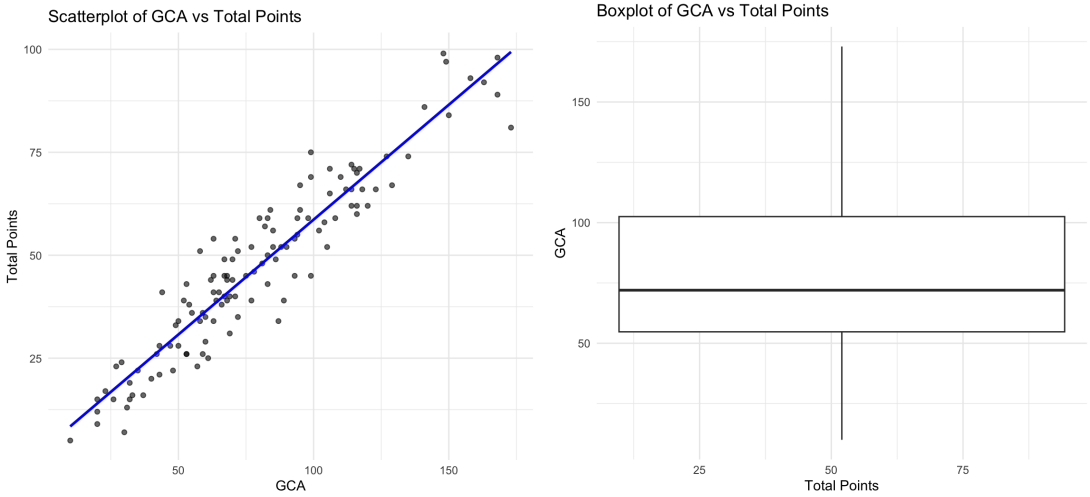
[1,] 0.274745 0.4610785 0.4637777 0.3841085 0.1171451

I then converted this into a vector, and sorted it in descending order. From there, I extracted the top ten metrics using head(). At this stage in the study, we have solved our first research question. The top ten metrics we identified in order were:.

1. Wins: Number of wins in a season.
2. Goals per 90: Goals scored every 90 minutes (full match length).
3. Goal Creating Actions: Goal Creating Actions (GCA) and Shot Creating Actions (SCA), meaning the two offensive leading to a shot or goal. This includes live-ball passes, dead-ball passes, successful dribbles, shots which lead to another shot, and being fouled.
4. Goals and Assists per 90: Goals and Assists every 90 minutes (full match length)
5. Non penalty Goals per 90: Goals and Assists every 90 minutes without goals from penalty kicks
6. Live Passes: completed live ball passes that led to a shot attempt.
7. Assists: Pass prior to a goal.
8. Expected Goals: Calculated metric that measures the probability of a given shot to become a goal, based on attacker and defender positioning.
9. Non-penalty expected goals: Calculated metric that measures the probability of a given shot to become a goal, based on attacker and defender positioning, with penalty kicks excluded.
10. Non-penalty expected goals and assists: Calculated metric that measures the probability of a given shot to become a goal, based on attacker and defender positioning, with penalty kicks excluded



To Visualize these findings, I used a boxplot, which shows the correlation between each metric and the total number of points. I also used box and scatter plots for further visualization.



To dive deeper into the analysis of the metrics, I did a multiple regression model that displays the total change in points that a change of one unit of each metric would affect.

```
Residuals:
    Min      1Q  Median      3Q     Max
-7.9835 -1.9136 -0.3058  1.9096  6.9080

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.09488    0.88730   2.361   0.0200 *
W            2.58212    0.12176  21.206   <2e-16 ***
Gls         -0.33290    0.37703  -0.883   0.3792
GCA          0.17879    0.11371   1.572   0.1187
G.A          0.01007    0.12950   0.078   0.9382
G.PK         0.10770    0.30951   0.348   0.7285
PassLive.1  -0.11293    0.06449  -1.751   0.0827 .
Ast               NA         NA      NA       NA
xG           0.42457    0.33571   1.265   0.2086
npxG         0.58349    0.44382   1.315   0.1913
npxG.xAG    -0.41884    0.19350  -2.165   0.0326 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.954 on 110 degrees of freedom
Multiple R-squared:  0.9821,    Adjusted R-squared:  0.9806
F-statistic: 668.9 on 9 and 110 DF,  p-value: < 2.2e-16
```

From this, we can see that Wins and Non-penalty Expected Goals and Assists have smaller p-values than 0.05, which means that those metrics are statistically significant.

Having discovered which metrics are the best indicators, we can now move on to predicting the winner of the English Premier League. To do this, I ranked each club for a given season, where they rank on each of the top metrics from one to twenty (total number of clubs per season). After this, I summed all the values, and sorted in ascending order. The top club, which would have the lowest total score, would be the predicted winner. I started with the 2018/19 data, which according to the top ten metrics, would have Manchester City as their winners. This, along with two other predictions were correct. The top ten metrics accurately predicted the winner 60% of the time. While this may not seem that accurate, there are certain caveats to mention. The first incorrect prediction, the 2019/20 season, was a season affected by covid. The season was stopped for several months after a decision on March 13 that suspended games until the middle of June, when games resumed behind closed doors (no fans were allowed in the stadium). These circumstances would obviously produce different results from the norm. This could possibly account for

the incorrect prediction. The other incorrect prediction, the 2021/22 season was a season where the title came to the last matchday, with the teams ending the season with one point between them. Manchester City ended the season on 93 points and Liverpool on 92 points. The total score according to the ten metrics I chose were 14 for Liverpool and 15 for Manchester City. The differences in results were razor thin both in real life and according to the chosen metrics for that season.

As for the current 2023/24 season, the chosen metrics have chosen Manchester City as the predicted winners, based on the statistics of the season to date. However, Manchester City are tied for first place with Newcastle, and have a significant gap between the two teams and the rest of the table. This proves Manchester City and Newcastle's exceptional performance in the league to this date, but obviously this form of prediction does not take into account the teams they played and the strength of competition faced.

Conclusion:

In conclusion, we successfully identified the top ten metrics that correlate to total points scored, and visualized our results in the form of box, scatter, and bar plots. Moreover, we ran a multiple regression model to measure the per unit change effect of our chosen metrics. The final portion of my project, the prediction, was quite accurate in several scenarios but for others had inconsistent results. Using these metrics alone as a predictor without having the context of strength of competition faced is certainly not the best measurement, but it does give an indication of how well teams have performed up to date.

Code

```
Python (extract data):
import pandas as pd
from google.colab import files
df = pd.read_html('https://fbref.com/en/comps/9/Premier-League-Stats')
df[16]
```

```
df[16].to_csv('Premier-League-DAdata-23.csv')
files.download('Premier-League-DAdata-23.csv')
```

R (main code):

```
library(ggplot2)

season_1819 <- read.csv("~/Downloads/Premier-league-data-18.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
season_1920 <- read.csv("~/Downloads/Premier-league-data-19.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
season_2021 <- read.csv("~/Downloads/Premier-league-data-20.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
season_2122 <- read.csv("~/Downloads/Premier-league-data-21.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
season_2223 <- read.csv("~/Downloads/Premier-league-data-22.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
season_2324 <- read.csv("~/Downloads/Premier-league-data-23.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)

wl_1819 <- read.csv("~/Downloads/Premier-League-WLdata-18.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]
wl_1920 <- read.csv("~/Downloads/Premier-League-WLdata-19.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]
wl_2021 <- read.csv("~/Downloads/Premier-League-WLdata-20.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]
wl_2122 <- read.csv("~/Downloads/Premier-League-WLdata-21.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]
wl_2223 <- read.csv("~/Downloads/Premier-League-WLdata-22.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]
```

```
wl_2324 <- read.csv("~/Downloads/Premier-League-WLdata-23.csv", stringsAsFactors = FALSE)[,
c("Squad", "W", "D", "L")]

da_1819 <- read.csv("~/Downloads/Premier-league-DAdata-18.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
da_1920 <- read.csv("~/Downloads/Premier-league-DAdata-19.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
da_2021 <- read.csv("~/Downloads/Premier-league-DAdata-20.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
da_2122 <- read.csv("~/Downloads/Premier-league-DAdata-21.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
da_2223 <- read.csv("~/Downloads/Premier-league-DAdata-22.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
da_2324 <- read.csv("~/Downloads/Premier-league-DAdata-23.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)

sca_1819 <- read.csv("~/Downloads/Premier-league-SCAdata-18.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
sca_1920 <- read.csv("~/Downloads/Premier-league-SCAdata-19.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
sca_2021 <- read.csv("~/Downloads/Premier-league-SCAdata-20.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
sca_2122 <- read.csv("~/Downloads/Premier-league-SCAdata-21.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
sca_2223 <- read.csv("~/Downloads/Premier-league-SCAdata-22.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)
sca_2324 <- read.csv("~/Downloads/Premier-league-SCAdata-23.csv", skip=1, header=TRUE,
stringsAsFactors = FALSE)

# Add the season data
wl_1819$Season <- "2018/19"
```

```
wl_1920$Season <- "2019/20"

wl_2021$Season <- "2020/21"

wl_2122$Season <- "2021/22"

wl_2223$Season <- "2022/23"

wl_2324$Season <- "2023/24"


season_1819$Season <- "2018/19"

season_1920$Season <- "2019/20"

season_2021$Season <- "2020/21"

season_2122$Season <- "2021/22"

season_2223$Season <- "2022/23"

season_2324$Season <- "2023/24"


da_1819$Season <- "2018/19"

da_1920$Season <- "2019/20"

da_2021$Season <- "2020/21"

da_2122$Season <- "2021/22"

da_2223$Season <- "2022/23"

da_2324$Season <- "2023/24"


sca_1819$Season <- "2018/19"

sca_1920$Season <- "2019/20"

sca_2021$Season <- "2020/21"

sca_2122$Season <- "2021/22"

sca_2223$Season <- "2022/23"

sca_2324$Season <- "2023/24"




# Combine all the WL data into one dataframe

wl_data <- rbind(wl_1819, wl_1920, wl_2021, wl_2122, wl_2223, wl_2324)
```

```
all_seasons_data <- rbind(season_1819, season_1920, season_2021, season_2122, season_2223,
season_2324)
all_da_data <- rbind(da_1819, da_1920, da_2021, da_2122, da_2223, da_2324)
all_sca_data <- rbind(sca_1819, sca_1920, sca_2021, sca_2122, sca_2223, sca_2324)



all_data <- merge(all_seasons_data, wl_data, by=c("Squad", "Season"))
all_data <- merge(all_data, all_da_data, by=c("Squad", "Season"))
all_data <- merge(all_data, all_sca_data, by=c("Squad", "Season"))

all_data$Pts <- all_data$W * 3 + all_data$D



numeric_columns <- sapply(all_data, is.numeric)
points_correlations <- cor(all_data$Pts, all_data[, numeric_columns], use = "complete.obs")
points_correlations <- points_correlations[, colnames(points_correlations) != "Pts", drop =
FALSE]
vec <- as.vector(points_correlations)
names(vec) <- colnames(points_correlations)
sorted_vec <- sort(vec, decreasing = TRUE)
top_values <- sorted_vec[1:10]
top_variables <- names(top_values)
print(winloss_correlations)
print(top_variables)
print(top_values)

plot_data <- data.frame(
  Metric = top_variables,
  Correlation = top_values
)
```

```
#correlation
correlation_plot <- ggplot(plot_data, aes(x = Metric, y = Correlation)) +
  geom_bar(stat = "identity", fill = "blue") +
  coord_flip() +
  labs(x = "Metric", y = "Correlation with Total Points",
       title = "Top 10 Metrics Correlated with Total Points") +
  theme_minimal()

print(correlation_plot)

#boxplots
for (metric in top_variables) {
  boxplot <- ggplot(all_data, aes_string(x = "Pts", y = metric)) +
    geom_boxplot() +
    labs(title = paste("Boxplot of", metric, "vs Total Points"),
         x = "Total Points",
         y = metric) +
    theme_minimal()

  print(boxplot)
}

#Scatterplots
for (metric in top_variables) {
  scatterplot <- ggplot(all_data, aes_string(x = metric, y = "Pts")) +
    geom_point(alpha = 0.6) +
    geom_smooth(method = "lm", se = FALSE, color = "blue") +
    labs(title = paste("Scatterplot of", metric, "vs Total Points"),
         x = metric,
         y = "Total Points") +
    theme_minimal()
```

```
  print(scatterplot)
}

formula_string <- paste("Pts ~", paste(top_variables, collapse = " + "))
model <- lm(formula_string, data = all_data)
print(summary(model))

par(mfrow = c(2, 2))
plot(model)

current_season_data <- all_data[all_data$Season == "2023/24", c("Squad", top_variables)]



ranked_data <- as.data.frame(lapply(current_season_data[, -1], function(x) rank(-x,
ties.method = "min")))
ranked_data$Squad <- current_season_data$Squad
ranked_data$TotalScore <- rowSums(ranked_data[, -ncol(ranked_data)])
ranked_data <- ranked_data[order(ranked_data$TotalScore), ]
print(ranked_data)
predicted_winner <- ranked_data$Squad[1]
print(paste("Predicted Premier League winner for the 2023/24 season is:", predicted_winner))
```