Final Project Report

IST687 - Group B2

9/26/2018

## Table of Contents

### INTRODUCTION

1. Project Background and Description
2. Project Scope and Context of this Analysis

### BUSINESS QUESTIONS

1. What are the Business Questions?

### DATA ACQUISITION, CLEANING, TRANSFORMATION, MUNGING

1. Describe your data acquisition process
2. What data did you select, all, subset, why
3. What was your initial quality assessment
4. What fields/variables did you finally decide on, why
5. Provide a data dictionary
6. Provide data descriptive statistics, rows, str
7. Did you have to do any cleansing, describe
8. Interesting findings

### DESCRIPTIVE STATISTICS

1. Provide demographic statistics – Location
2. Any early observations, nuggets of interest, interpretation, interesting findings

### USE OF MODELING TECHNIQUES

1. Modeling approach
2. Classification models
3. Plotting and further interpreting results

### CONCLUSIONS

1. Final thoughts

### OVERALL INTERPRETATION OF RESULTS/ACTIONABLE INSIGHTS

### REFERENCES

### APPENDIX – RStudio CODE

# FINAL PROJECT REPORT

## IST687 - GROUP B2

### INTRODUCTION

1. Project Background and Description  
   This project is an exercise in taking a dataset from the 2017-2018 English Premier League season and tries to gauge what factors are able to predict which team is likelier to win; what factors are likely to separate winning teams from losing teams, and what can teams from the bottom of the chart learn from teams at the top.
2. Project Scope and Context of this Analysis  
   The scope of this project encompasses data gathered from the 2017-2018 English Premier League season, with goals scored, goals allowed, shots taken, shots allowed, win record, and other soccer metrics. The data shows whether a team won, lost, or drew a game, how many shots did they take and allow, how many fouls they commited, how many goals they scored (both at half-time and end of regular time), and whether they received any yellow or red cards. Obviously, we expect that teams that score more goals tend to win more but, by looking at actual goal differential, we might find that this is not the case. We also want to see whether there are other factors that are pushing a teams’ likelihood to win - such as shots taken, shots on goal taken, and corner kicks - as against to allowing an opposing team more chances to score.

### BUSINESS QUESTIONS

1. What are the Business Questions?  
   There are three main questions we want to address with this project:
2. Is there home-field advantage in the English Premier League?
3. Does the score at half-time affect the final outcome of the game?
4. Are the number of fouls, cards, and shots correlated to the final score of the game?

### DATA ACQUISITION, CLEANSING, TRANSFORMATION, AND MUNGING

1. Describe your data acquisition process.  
   The data was acquired from www.football-data.co.uk, a website that provides free data on soccer coverage from around the world. For this project, we downloaded the csv file for the English Premier League 2017-2018 season. We grabbed the csv url and loaded it into RStudio using the following command:

## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character

## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector

## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character

## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector

## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character

## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector

## Warning in bind\_rows\_(x, .id): Unequal factor levels: coercing to character

## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector

## Div Date HomeTeam AwayTeam FTHG FTAG FTR HTHG HTAG HTR  
## 1 E0 08/08/15 Bournemouth Aston Villa 0 1 A 0 0 D  
## 2 E0 08/08/15 Chelsea Swansea 2 2 D 2 1 H  
## 3 E0 08/08/15 Everton Watford 2 2 D 0 1 A  
## 4 E0 08/08/15 Leicester Sunderland 4 2 H 3 0 H  
## 5 E0 08/08/15 Man United Tottenham 1 0 H 1 0 H  
## 6 E0 08/08/15 Norwich Crystal Palace 1 3 A 0 1 A  
## Referee HS AS HST AST HF AF HC AC HY AY HR AR B365H B365D B365A  
## 1 M Clattenburg 11 7 2 3 13 13 6 3 3 4 0 0 2.00 3.6 4.00  
## 2 M Oliver 11 18 3 10 15 16 4 8 1 3 1 0 1.36 5.0 11.00  
## 3 M Jones 10 11 5 5 7 13 8 2 1 2 0 0 1.70 3.9 5.50  
## 4 L Mason 19 10 8 5 13 17 6 3 2 4 0 0 1.95 3.5 4.33  
## 5 J Moss 9 9 1 4 12 12 1 2 2 3 0 0 1.65 4.0 6.00  
## 6 S Hooper 17 11 6 7 14 20 1 4 1 0 0 0 2.55 3.3 3.00  
## BWH BWD BWA IWH IWD IWA LBH LBD LBA PSH PSD PSA WHH WHD  
## 1 2.00 3.30 3.70 2.10 3.3 3.30 2.05 3.3 4.0 1.95 3.65 4.27 1.91 3.5  
## 2 1.40 4.75 9.00 1.33 4.8 8.30 1.40 4.5 10.0 1.39 4.92 10.39 1.40 4.0  
## 3 1.70 3.50 5.00 1.70 3.6 4.70 1.75 3.8 5.0 1.70 3.95 5.62 1.73 3.5  
## 4 2.00 3.30 3.75 2.00 3.3 3.60 2.00 3.4 4.2 1.99 3.48 4.34 2.00 3.1  
## 5 1.65 4.00 5.50 1.65 3.6 5.10 1.67 4.0 5.5 1.65 4.09 5.90 1.62 3.6  
## 6 2.60 3.20 2.70 2.40 3.2 2.85 2.62 3.2 2.9 2.52 3.35 3.08 2.60 3.1  
## WHA VCH VCD VCA Bb1X2 BbMxH BbAvH BbMxD BbAvD BbMxA BbAvA BbOU  
## 1 4.00 2.00 3.50 4.20 45 2.10 1.96 3.65 3.48 4.33 3.98 43  
## 2 10.00 1.40 5.00 9.50 45 1.43 1.37 5.00 4.66 11.26 9.57 43  
## 3 5.00 1.73 3.90 5.40 45 1.75 1.69 4.00 3.76 5.77 5.25 44  
## 4 2.70 2.00 3.40 4.33 45 2.03 1.96 3.50 3.37 4.52 4.06 43  
## 5 6.00 1.67 4.00 5.75 42 1.71 1.63 4.20 3.90 6.50 5.65 40  
## 6 2.88 2.60 3.25 3.00 45 2.63 2.53 3.42 3.21 3.11 2.92 43  
## BbMx.2.5 BbAv.2.5 BbMx.2.5.1 BbAv.2.5.1 BbAH BbAHh BbMxAHH BbAvAHH  
## 1 2.11 2.02 1.88 1.79 26 -0.5 1.98 1.93  
## 2 1.88 1.80 2.07 1.99 27 -1.5 2.24 2.16  
## 3 1.93 1.84 2.03 1.96 26 -1.0 2.28 2.18  
## 4 2.27 2.18 1.73 1.67 26 -0.5 2.00 1.95  
## 5 1.86 1.79 2.13 2.01 26 -1.0 2.20 2.09  
## 6 2.25 2.18 1.75 1.67 27 0.0 1.83 1.78  
## BbMxAHA BbAvAHA PSCH PSCD PSCA  
## 1 1.99 1.92 1.82 3.88 4.70  
## 2 1.80 1.73 1.37 5.04 10.88  
## 3 1.76 1.71 1.75 3.76 5.44  
## 4 1.96 1.90 1.79 3.74 5.10  
## 5 1.82 1.78 1.64 4.07 6.04  
## 6 2.17 2.08 2.46 3.39 3.14

1. What data did you select, all, subset, why?  
   We decided to focus on just a single season of data, selecting data for both the home and away team regarding whether the home team one, lost, or drew, how many goals each team scored both at half-time and end of regulation, how many shots each team took both in general and on goal, corner kicks, fouls committed, yellow and red cards received, and other data such as the date, and referee. We also had different betting odds for all teams and for each match but we decided to drop these columns as they weren’t part of the scope of the project.
2. What was your initial quality assessment?  
   Our first quality assessment was to check that there were no missing values within the dataset.  
   After that, we made sure that the data types were correct, i.e. strings weren’t stored as factors, numbers were numeric, and dates were stored as dates.  
   We also made sure that there were no NULL values or NA's, which fortunately there weren’t.  
   Finally, after the assessment was made, we decided to transform the dataset in order to have one row per team per date, renaming the columns, dropping the betting line columns, and adding a flag to signal whether the team in question was the home or away team.

## Div Date Team1 Team2 HomeAway GS GA FTR HTGS HTGA HTR  
## 1 E0 08/08/15 Bournemouth Aston Villa Home 0 1 A 0 0 D  
## 2 E0 08/08/15 Chelsea Swansea Home 2 2 D 2 1 H  
## 3 E0 08/08/15 Everton Watford Home 2 2 D 0 1 A  
## 4 E0 08/08/15 Leicester Sunderland Home 4 2 H 3 0 H  
## 5 E0 08/08/15 Man United Tottenham Home 1 0 H 1 0 H  
## 6 E0 08/08/15 Norwich Crystal Palace Home 1 3 A 0 1 A  
## Referee ShotsTaken ShotsAllowed ShotsOnTarget OppShotsOnTarget  
## 1 M Clattenburg 11 7 2 3  
## 2 M Oliver 11 18 3 10  
## 3 M Jones 10 11 5 5  
## 4 L Mason 19 10 8 5  
## 5 J Moss 9 9 1 4  
## 6 S Hooper 17 11 6 7  
## FoulsCommitted FoulsReceived CornerKicks OppCornerKicks YellowCards  
## 1 13 13 6 3 3  
## 2 15 16 4 8 1  
## 3 7 13 8 2 1  
## 4 13 17 6 3 2  
## 5 12 12 1 2 2  
## 6 14 20 1 4 1  
## OppYellowCards RedCards OppRedCards WinOdds DrawOdds LossOdds  
## 1 4 0 0 2.00 3.6 4.00  
## 2 3 1 0 1.36 5.0 11.00  
## 3 2 0 0 1.70 3.9 5.50  
## 4 4 0 0 1.95 3.5 4.33  
## 5 3 0 0 1.65 4.0 6.00  
## 6 0 0 0 2.55 3.3 3.00

1. What fields/variables did you finally decide on, why  
   The original dataset had 65 columns and 380 rows. The variables included regarded the league, date, teams playing, referee, match results and goals at half-time and at the end of regulation, statistics for the match, and betting odds for different sites. We transformed this data so as to end with 24 columns and 780 rows, setting on the following columns:

* Match Information: Div, Date, Team1, Team2, HomeAway, Referee
* Match Results: GS, GA, FTR, HTGS, HTGA, HTR
* Match Statistics: ShotsTaken, ShotsAllowed, ShotsOnTarget, OppShotsOnTarget, FoulsCommitted, FoulsReceived, CornerKicks, OppCornerKicks, YellowCards, OppYellowCards, RedCards, OppRedCards

colnames(premTransformed)

## [1] "Div" "Date" "Team1"   
## [4] "Team2" "HomeAway" "GS"   
## [7] "GA" "FTR" "HTGS"   
## [10] "HTGA" "HTR" "Referee"   
## [13] "ShotsTaken" "ShotsAllowed" "ShotsOnTarget"   
## [16] "OppShotsOnTarget" "FoulsCommitted" "FoulsReceived"   
## [19] "CornerKicks" "OppCornerKicks" "YellowCards"   
## [22] "OppYellowCards" "RedCards" "OppRedCards"   
## [25] "WinOdds" "DrawOdds" "LossOdds"

1. Provide a data dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Index | Column Name | New Names | Definition |
| 1 | Div | Division | Football league division. |
| 2 | Date | Match Day | Day match was played. |
| 3 | Team1 | Team | First team of the match. |
| 4 | Team2 | Opponent | Opposing team of the match. |
| 5 | HomeAway | Home-Away Flag | Team1 Home-Away identifier. |
| 6 | GS | Goals Scored | Goals scored by Team1. |
| 7 | GA | Goals Allowed | Goals allowed by Team1. |
| 8 | FTR | Full Time Result | Result at the end of regulation. |
| 9 | HTGS | Half-time Goals Scored | Goals scored at half-time by Team1. |
| 10 | HTGA | Half-time Goals Allowed | Goals allowed at half-time by Team1. |
| 11 | HTR | Half-time Result | Result at half-time. |
| 12 | Referee | Game Referee | Referee for the match. |
| 13 | ShotsTaken | Shots taken | Shots taken by Team 1. |
| 14 | ShotsAllowed | Shots allowed | Shots allowed by Team 1. |
| 15 | ShotsOnTarget | Shots on target | Shots on goal taken by Team 1. |
| 16 | OppShotsOnTarget | Shots on target allowed | Shots on goal allowed by Team 1. |
| 17 | FoulsCommitted | Fouls committed | Fouls committed by Team 1. |
| 18 | FoulsReceived | Fouls received | Fouls committed by Opposing Team. |
| 19 | CornerKicks | Corner kicks | Corner kicks taken by Team 1. |
| 20 | OppCornerKicks | Opponent corner kicks | Corner kicks taken by Opposing Team. |
| 21 | YellowCards | Yellow cards | Yellow cards given to Team 1. |
| 22 | OppYellowCards | Opponent yellow cards | Yellow cards given to Opposing Team. |
| 23 | RedCards | Red cards | Red cards given to Team 1. |
| 24 | OppRedCards | Opponent red cards | Red cards given to Opposing Team. |

1. Provide data descriptive statistics, rows, str  
   The complete dataset has 24 variables and 780 observations - two per match. The dataset does not have any NA’s, meaning that there will be no additional data dropped.  
   The following shows the dimensions and structure of the dataset.

## [1] 2280 27

## 'data.frame': 2280 obs. of 27 variables:  
## $ Div : Factor w/ 1 level "E0": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Date : chr "08/08/15" "08/08/15" "08/08/15" "08/08/15" ...  
## $ Team1 : chr "Bournemouth" "Chelsea" "Everton" "Leicester" ...  
## $ Team2 : chr "Aston Villa" "Swansea" "Watford" "Sunderland" ...  
## $ HomeAway : chr "Home" "Home" "Home" "Home" ...  
## $ GS : int 0 2 2 4 1 1 0 2 0 0 ...  
## $ GA : int 1 2 2 2 0 3 2 2 1 3 ...  
## $ FTR : Factor w/ 3 levels "A","D","H": 1 2 2 3 3 1 1 2 1 1 ...  
## $ HTGS : int 0 2 0 3 1 0 0 1 0 0 ...  
## $ HTGA : int 0 1 1 0 0 1 1 1 0 2 ...  
## $ HTR : Factor w/ 3 levels "A","D","H": 2 3 1 3 3 1 1 2 2 1 ...  
## $ Referee : chr "M Clattenburg" "M Oliver" "M Jones" "L Mason" ...  
## $ ShotsTaken : int 11 11 10 19 9 17 22 9 7 9 ...  
## $ ShotsAllowed : int 7 18 11 10 9 11 8 15 8 19 ...  
## $ ShotsOnTarget : int 2 3 5 8 1 6 6 4 1 2 ...  
## $ OppShotsOnTarget: int 3 10 5 5 4 7 4 5 3 7 ...  
## $ FoulsCommitted : int 13 15 7 13 12 14 12 9 9 12 ...  
## $ FoulsReceived : int 13 16 13 17 12 20 9 12 16 9 ...  
## $ CornerKicks : int 6 4 8 6 1 1 5 6 3 6 ...  
## $ OppCornerKicks : int 3 8 2 3 2 4 4 6 5 6 ...  
## $ YellowCards : int 3 1 1 2 2 1 1 2 2 4 ...  
## $ OppYellowCards : int 4 3 2 4 3 0 3 4 4 1 ...  
## $ RedCards : int 0 1 0 0 0 0 0 0 0 0 ...  
## $ OppRedCards : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ WinOdds : num 2 1.36 1.7 1.95 1.65 2.55 1.29 2.88 3.4 5.75 ...  
## $ DrawOdds : num 3.6 5 3.9 3.5 4 3.3 6 3.3 3.4 4 ...  
## $ LossOdds : num 4 11 5.5 4.33 6 3 12 2.7 2.3 1.67 ...

1. Did you have to do any cleansing, describe  
   No, the dataset was already cleaned when we downloaded it.  
   However, we did need to transform it as the data wasn’t in the format we desired. We transformed it so that we our analysis and data modeling would be better adjusted and easier for the later stages of the project.
2. Interesting findings  
   What was initially interesting is that there seems to be no home-field advantage. Teams are likely to win at home or away, which speaks to the competitiveness of the league. We also found that, although red cards are significant when determining the outcome of the game, yellow cards aren’t; which is interesting as the number of yellow cards could lead to substitutions or teams playing more calmly. A third interesting finding was that there is a forced variation each season because of teams being relegated and promoted from the Second Division - something that we saw in the data but weren’t able to study because it was outside the scope of the project.

### DESCRIPTIVE STATISTICS

1. Provide demographic statistics – Location  
   The English Premier League constitutes the highest level of professional football in England. Contested by 20 clubs, it operates on a system of promotion and relegation with the English Football League (EFL). Seasons run from August to May with each team playing 38 matches (playing each other home and away). As illustrated in the below map, the 20 teams 20 for the season 2017-2018 were widely dispersed geographically.  
   In the descriptive analysis of the Premier League dataset the focus was on the followings:  
   • Measures of Central Tendency and Dispersion: the mean, standard deviation, median, minimum and maximum values, as well as quantiles were calculated. Below table summarizes this information for one of the key variables i.e. Full Time Win.

## [1] "Bournemouth" "Chelsea" "Everton" "Leicester"   
## [5] "ManUnited" "Norwich" "Arsenal" "Newcastle"   
## [9] "Stoke" "WestBrom" "AstonVilla" "Southampton"   
## [13] "Sunderland" "Swansea" "Tottenham" "Watford"   
## [17] "WestHam" "CrystalPalace" "ManCity" "Liverpool"   
## [21] "Burnley" "Hull" "Middlesbrough" "Brighton"   
## [25] "Huddersfield"

## Team1 HomeAway TotalShotsTaken TotalShotsOnTarget  
## 1 Bournemouth Home 763 256  
## 2 Bournemouth Away 612 205  
## 3 Chelsea Home 974 346  
## 4 Chelsea Away 738 251  
## 5 Everton Home 745 266  
## 6 Everton Away 605 223  
## 7 Leicester Home 743 247  
## 8 Leicester Away 634 227  
## 9 ManUnited Home 0 0  
## 10 ManUnited Away 0 0  
## 11 Norwich Home 232 68  
## 12 Norwich Away 184 58  
## 13 Arsenal Home 993 375  
## 14 Arsenal Away 735 268  
## 15 Newcastle Home 452 162  
## 16 Newcastle Away 395 140  
## 17 Stoke Home 651 208  
## 18 Stoke Away 575 190  
## 19 WestBrom Home 0 0  
## 20 WestBrom Away 0 0  
## 21 AstonVilla Home 0 0  
## 22 AstonVilla Away 0 0  
## 23 Southampton Home 837 263  
## 24 Southampton Away 681 230  
## 25 Sunderland Home 462 138  
## 26 Sunderland Away 364 126  
## 27 Swansea Home 665 214  
## 28 Swansea Away 503 172  
## 29 Tottenham Home 1097 412  
## 30 Tottenham Away 855 312  
## 31 Watford Home 717 221  
## 32 Watford Away 587 191  
## 33 WestHam Home 0 0  
## 34 WestHam Away 0 0  
## 35 CrystalPalace Home 0 0  
## 36 CrystalPalace Away 0 0  
## 37 ManCity Home 0 0  
## 38 ManCity Away 0 0  
## 39 Liverpool Home 1059 370  
## 40 Liverpool Away 848 302  
## 41 Burnley Home 417 130  
## 42 Burnley Away 352 117  
## 43 Hull Home 208 69  
## 44 Hull Away 188 58  
## 45 Middlesbrough Home 205 59  
## 46 Middlesbrough Away 144 41  
## 47 Brighton Home 217 71  
## 48 Brighton Away 166 47  
## 49 Huddersfield Home 207 55  
## 50 Huddersfield Away 153 56  
## TotalFoulsCommitted TotalCornerKicks TotalYellowCards TotalRedCards  
## 1 513 342 80 2  
## 2 552 287 78 3  
## 3 576 404 78 6  
## 4 589 284 93 4  
## 5 595 325 87 3  
## 6 638 239 79 7  
## 7 560 313 69 3  
## 8 661 284 102 6  
## 9 0 0 0 0  
## 10 0 0 0 0  
## 11 201 108 29 0  
## 12 216 80 32 3  
## 13 556 396 79 5  
## 14 574 281 84 4  
## 15 409 160 60 4  
## 16 414 165 50 3  
## 17 627 271 89 5  
## 18 662 206 94 2  
## 19 0 0 0 0  
## 20 0 0 0 0  
## 21 0 0 0 0  
## 22 0 0 0 0  
## 23 588 347 79 5  
## 24 668 304 99 5  
## 25 395 185 64 3  
## 26 438 135 74 3  
## 27 559 291 89 1  
## 28 611 218 80 1  
## 29 602 433 77 0  
## 30 666 340 106 2  
## 31 707 273 129 6  
## 32 711 241 91 6  
## 33 0 0 0 0  
## 34 0 0 0 0  
## 35 0 0 0 0  
## 36 0 0 0 0  
## 37 0 0 0 0  
## 38 0 0 0 0  
## 39 540 392 63 1  
## 40 623 352 96 3  
## 41 375 173 62 0  
## 42 401 143 68 2  
## 43 219 73 32 2  
## 44 184 106 35 3  
## 45 210 93 38 0  
## 46 268 48 38 1  
## 47 198 90 37 1  
## 48 215 73 17 1  
## 49 180 102 26 2  
## 50 216 63 35 1  
## TotalFTGoals TotalHTGoals TotalHTWin TotalHTLoss TotalHTDraw TotalFTWin  
## 1 84 34 16 20 21 21  
## 2 61 31 22 10 25 28  
## 3 117 54 25 5 27 33  
## 4 89 42 16 25 16 16  
## 5 105 41 18 12 27 29  
## 6 60 27 21 11 25 24  
## 7 91 40 24 14 19 29  
## 8 81 37 19 15 23 24  
## 9 0 0 0 0 0 0  
## 10 0 0 0 0 0 0  
## 11 26 10 5 6 8 6  
## 12 13 6 7 2 10 14  
## 13 124 55 26 10 21 41  
## 14 92 37 13 16 28 22  
## 15 53 22 11 9 18 15  
## 16 30 11 18 5 15 25  
## 17 66 32 18 17 22 20  
## 18 51 25 22 15 20 28  
## 19 0 0 0 0 0 0  
## 20 0 0 0 0 0 0  
## 21 0 0 0 0 0 0  
## 22 0 0 0 0 0 0  
## 23 76 34 19 14 24 21  
## 24 61 27 17 14 26 23  
## 25 39 15 7 14 17 9  
## 26 38 14 16 4 18 25  
## 27 64 30 17 17 23 22  
## 28 51 20 24 9 24 34  
## 29 122 52 27 8 22 40  
## 30 107 42 14 17 26 12  
## 31 72 27 14 17 26 21  
## 32 52 25 24 12 21 36  
## 33 0 0 0 0 0 0  
## 34 0 0 0 0 0 0  
## 35 0 0 0 0 0 0  
## 36 0 0 0 0 0 0  
## 37 0 0 0 0 0 0  
## 38 0 0 0 0 0 0  
## 39 123 55 27 5 25 32  
## 40 102 47 16 22 19 16  
## 41 42 19 11 6 21 17  
## 42 33 13 12 6 20 19  
## 43 28 8 4 6 9 8  
## 44 9 3 10 1 8 15  
## 45 17 9 5 7 7 4  
## 46 10 6 8 1 10 11  
## 47 24 8 3 4 12 7  
## 48 10 8 8 4 7 12  
## 49 16 10 6 6 7 6  
## 50 12 5 5 2 12 11  
## TotalFTLoss TotalFTDraw TotalGamesPlayed  
## 1 22 14 57  
## 2 13 16 57  
## 3 11 13 57  
## 4 30 11 57  
## 5 15 13 57  
## 6 12 21 57  
## 7 12 16 57  
## 8 18 15 57  
## 9 0 0 0  
## 10 0 0 0  
## 11 8 5 19  
## 12 3 2 19  
## 13 7 9 57  
## 14 21 14 57  
## 15 12 11 38  
## 16 6 7 38  
## 17 22 15 57  
## 18 12 17 57  
## 19 0 0 0  
## 20 0 0 0  
## 21 0 0 0  
## 22 0 0 0  
## 23 20 16 57  
## 24 16 18 57  
## 25 18 11 38  
## 26 6 7 38  
## 27 23 12 57  
## 28 10 13 57  
## 29 5 12 57  
## 30 28 17 57  
## 31 20 16 57  
## 32 13 8 57  
## 33 0 0 0  
## 34 0 0 0  
## 35 0 0 0  
## 36 0 0 0  
## 37 0 0 0  
## 38 0 0 0  
## 39 5 20 57  
## 40 27 14 57  
## 41 13 8 38  
## 42 8 11 38  
## 43 7 4 19  
## 44 1 3 19  
## 45 9 6 19  
## 46 1 7 19  
## 47 4 8 19  
## 48 2 5 19  
## 49 8 5 19  
## 50 3 5 19

## [1] 13.9701754 4.6894737 5.7885965 1.5403509 0.6622807

## [1] 11.2359649 3.8333333 4.7219298 1.1859649 0.5184211

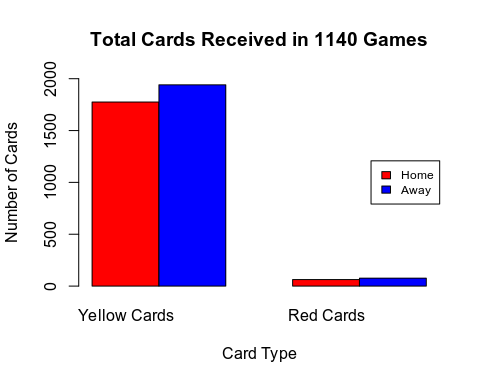
## TotalGamesPlayed TotalFoulsCommitted TotalFoulsReceived TotalYellowCards  
## 1 1140 11909 12768 1775  
## TotalRedCards TotalOppYellowCards TotalOppRedCards MeanFoulsCommitted  
## 1 63 1941 76 10.4  
## MeanFoulsReceived MeanYellowCards MeanOppYellowCards MeanRedCards  
## 1 11.2 1.6 1.7 0.1  
## MeanOppRedCards  
## 1 0.1

## 'data.frame': 2 obs. of 21 variables:  
## $ MeanFTWin : num 15.5 15.5  
## $ MeanFTLoss : num 9.42 9.42  
## $ MeanFTDraw : num 8.5 8.5  
## $ STDFTWin : num 12.2 12.2  
## $ STDFTLoss : num 8.67 8.67  
## $ STDFTDraw : num 6.48 6.48  
## $ MedianFTWin : num 15.5 15.5  
## $ MedianFTLoss : num 8 8  
## $ MedianFTDraw : num 8 8  
## $ MinFTWin : int 0 0  
## $ MaxFTWin : int 41 41  
## $ MinFTLoss : int 0 0  
## $ MaxFTLoss : int 30 30  
## $ MinFTDraw : int 0 0  
## $ MaxFTDraw : int 21 21  
## $ QuantilesFTWinInfo : num 0 35.1  
## $ SkewnessFTWinInfo : num 0.215 0.215  
## $ QuantilesFTLossInfo: num 0 25.2  
## $ SkewnessFTLossInfo : num 0.624 0.624  
## $ QuantilesFTDrawInfo: num 0 17.5  
## $ SkewnessFTDrawInfo : num 0.0455 0.0455

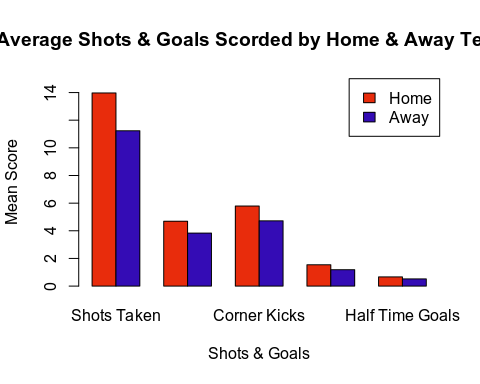
Moreover, the normal distribution was plotted for the following variables Full Time Win, Full Time Loss, and Full Time Draw.

The Full Time Win and Full Time Draw variables are normally distributed as they are symmetrical to some extent and the mean and median are very close. The Full Time Win and Full Time Draw are both slightly left-skewed as the median is larger than the mean.

• Team’s discipline: was measured by number of fouls committed, and number of yellow cards and red cards received. The figure below shows the number of yellow and red cards received by Home and Away teams. The Home teams were more disciplined.

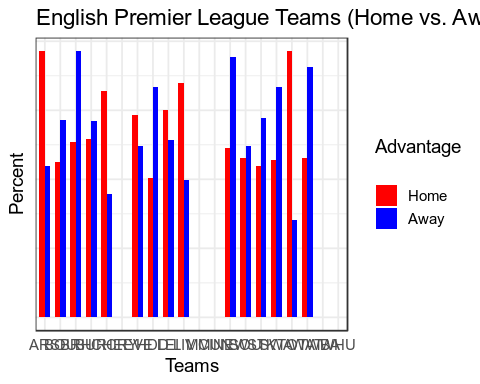


• Scoring Goals and Performance: the average of shots taken, and goals scored for and against Home and Away teams was calculated. The figure below demonstrates a comparison between Home and Away teams. The Home teams performed better as they had higher number of shots and corner kicks, and higher goals during half as well as full time.



• Home Advantage: was measured by calculating the points earned from all matches played at home versus away. The team gets 3 points for a win, 1 point for a draw, and zero for a loss. If the ratio is 50%, it indicates that success has been achieved equally at home and away i.e. there is no home field advantage. If the ratio is above 50% then it indicates that playing at home field increases the winning probability.

## Warning: Removed 10 rows containing missing values (geom\_bar).



1. Any early observations, nuggets of interest, interpretation, interesting findings  
   Overall, the findings were not surprising as they were consistent with our assumptions with the only exception of home advantage. Only 10 teams had a home advantage greater than 50%.

### USE OF MODELING TECHNIQUES

1. Modeling approach After loading the data 3 variables were added to the dataset; • Dependent Variable WIN:

premTransformed$win <- ifelse(premTransformed$GS > premTransformed$GA, 1, 0)  
# Note: a tie is categorized as a 0 in this formula and prediction

• Halftime Goal Difference:

premTransformed$HTdiff <- premTransformed$HTGS - premTransformed$HTGA

• Combine Yellow and Red cards into a single variable:

premTransformed$red\_yellow <- premTransformed$RedCards + premTransformed$YellowCards  
premTransformed$opp\_red\_yellow <- premTransformed$OppRedCards + premTransformed$OppYellowCards

• And transformed HomeAway to factor:

premTransformed$HomeAway <- as.factor(premTransformed$HomeAway)

We used a logistic regression model, where we first built the model using all the variables.

##   
## Call:  
## glm(formula = win ~ HomeAway + FTR + HTdiff + HTR + ShotsTaken +   
## ShotsAllowed + ShotsOnTarget + OppShotsOnTarget + FoulsCommitted +   
## FoulsReceived + CornerKicks + OppCornerKicks + YellowCards +   
## OppYellowCards + RedCards + OppRedCards + DrawOdds + LossOdds +   
## red\_yellow + opp\_red\_yellow, family = binomial(logit), data = premTransformed)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.12820 -0.22441 -0.00004 0.22441 3.12820   
##   
## Coefficients: (2 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.669e-01 6.939e-01 -0.385 0.70046   
## HomeAwayHome 5.339e-01 1.652e-01 3.232 0.00123 \*\*   
## FTRD -2.075e+01 3.794e+02 -0.055 0.95639   
## FTRH 2.907e-15 1.790e-01 0.000 1.00000   
## HTdiff 1.812e+00 1.208e-01 15.008 < 2e-16 \*\*\*  
## HTRD 3.770e-15 2.147e-01 0.000 1.00000   
## HTRH 1.119e-16 2.714e-01 0.000 1.00000   
## ShotsTaken 3.206e-02 2.442e-02 1.313 0.18928   
## ShotsAllowed -3.206e-02 2.442e-02 -1.313 0.18928   
## ShotsOnTarget 4.014e-01 5.104e-02 7.865 3.69e-15 \*\*\*  
## OppShotsOnTarget -4.014e-01 5.104e-02 -7.865 3.69e-15 \*\*\*  
## FoulsCommitted -2.553e-02 2.579e-02 -0.990 0.32228   
## FoulsReceived 2.553e-02 2.579e-02 0.990 0.32228   
## CornerKicks -8.994e-02 3.210e-02 -2.802 0.00508 \*\*   
## OppCornerKicks 8.994e-02 3.210e-02 2.802 0.00508 \*\*   
## YellowCards 5.692e-02 7.092e-02 0.803 0.42219   
## OppYellowCards -5.692e-02 7.092e-02 -0.803 0.42219   
## RedCards -9.522e-01 3.351e-01 -2.842 0.00449 \*\*   
## OppRedCards 9.522e-01 3.351e-01 2.842 0.00449 \*\*   
## DrawOdds 4.865e-16 1.143e-01 0.000 1.00000   
## LossOdds -1.181e-16 3.607e-02 0.000 1.00000   
## red\_yellow NA NA NA NA   
## opp\_red\_yellow NA NA NA NA   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3011.6 on 2279 degrees of freedom  
## Residual deviance: 1029.4 on 2259 degrees of freedom  
## AIC: 1071.4  
##   
## Number of Fisher Scoring iterations: 18

With this initial logistic regression done, we now filter down to the most significant variables.

##   
## Call:  
## glm(formula = win ~ HTdiff + ShotsOnTarget + OppShotsOnTarget +   
## CornerKicks + OppCornerKicks + RedCards + OppRedCards, family = binomial(logit),   
## data = premTransformed)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5309 -0.6432 -0.2324 0.5628 3.0943   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.91455 0.22293 -4.102 4.09e-05 \*\*\*  
## HTdiff 1.40544 0.08182 17.177 < 2e-16 \*\*\*  
## ShotsOnTarget 0.32908 0.02863 11.493 < 2e-16 \*\*\*  
## OppShotsOnTarget -0.29817 0.03112 -9.581 < 2e-16 \*\*\*  
## CornerKicks -0.06148 0.02199 -2.796 0.00517 \*\*   
## OppCornerKicks 0.03382 0.02209 1.531 0.12577   
## RedCards -0.75492 0.27436 -2.752 0.00593 \*\*   
## OppRedCards 0.73466 0.22827 3.218 0.00129 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3011.6 on 2279 degrees of freedom  
## Residual deviance: 1828.4 on 2272 degrees of freedom  
## AIC: 1844.4  
##   
## Number of Fisher Scoring iterations: 6

Our results have improved but they are still not at a point where we feel comfortable with them. We will do one final iteration.

##   
## Call:  
## glm(formula = win ~ GS + HTdiff + OppShotsOnTarget + RedCards,   
## family = binomial(logit), data = premTransformed)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.9310 -0.4030 -0.1092 0.2501 2.5249   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.41186 0.16498 -8.558 <2e-16 \*\*\*  
## GS 1.96932 0.10204 19.300 <2e-16 \*\*\*  
## HTdiff 0.99008 0.09113 10.864 <2e-16 \*\*\*  
## OppShotsOnTarget -0.52898 0.04037 -13.103 <2e-16 \*\*\*  
## RedCards -0.75586 0.32177 -2.349 0.0188 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3011.6 on 2279 degrees of freedom  
## Residual deviance: 1297.0 on 2275 degrees of freedom  
## AIC: 1307  
##   
## Number of Fisher Scoring iterations: 6

We have a good AIC of 1307 and find that by reducing to only four variables - GS, HTdiff, OppShotsOnTarget, and RedCards - we are able to make more accurate predictions. We will now test the accuracy of this model furthermore by training and testing it.

1. Classification Models We will test the four most significant variables by using the KSVM, SVM, and Naive Bayes algorithms to predict whether a team will win a game or lose/draw it.

The first step will be to create a training and test sets.

randIndex <- sample(1:dim(premTransformed)[1])  
# selects the cut off point for 67 percent of the data.  
cutPoint2\_3 <- floor(2 \* dim(premTransformed)[1]/3)  
  
# creates training dataset.  
trainData <- premTransformed[randIndex[1:cutPoint2\_3], ]  
# creates test dataset.  
testData <- premTransformed[randIndex[(cutPoint2\_3 + 1):dim(premTransformed)[1]],   
 ]  
  
# Convert the response variable to factor.  
trainData$win <- as.factor(trainData$win)  
testData$win <- as.factor(testData$win)

Now, we will start training and testing the algorithms, starting with the KSVM algorithm.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 430 53  
## 1 57 220  
##   
## Accuracy : 0.8553   
## 95% CI : (0.8282, 0.8795)  
## No Information Rate : 0.6408   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.6866   
## Mcnemar's Test P-Value : 0.7748   
##   
## Sensitivity : 0.8830   
## Specificity : 0.8059   
## Pos Pred Value : 0.8903   
## Neg Pred Value : 0.7942   
## Prevalence : 0.6408   
## Detection Rate : 0.5658   
## Detection Prevalence : 0.6355   
## Balanced Accuracy : 0.8444   
##   
## 'Positive' Class : 0   
##

KSVM gave us good results (84.3 percent accuracy!) Let’s try working with the SVM algorithm.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 176 36  
## 1 34 137  
##   
## Accuracy : 0.8172   
## 95% CI : (0.7748, 0.8547)  
## No Information Rate : 0.5483   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.6306   
## Mcnemar's Test P-Value : 0.9049   
##   
## Sensitivity : 0.8381   
## Specificity : 0.7919   
## Pos Pred Value : 0.8302   
## Neg Pred Value : 0.8012   
## Prevalence : 0.5483   
## Detection Rate : 0.4595   
## Detection Prevalence : 0.5535   
## Balanced Accuracy : 0.8150   
##   
## 'Positive' Class : 0   
##

We had an issue with the test dataset where only 392 of the 760 variables were being returned. Nevertheless, we filtered out those that weren’t being predicted and ended up with an 80.4 percent accuracy rate. Good but not an improvement on the KSVM algorithm. We will try running a Naive Bayes.

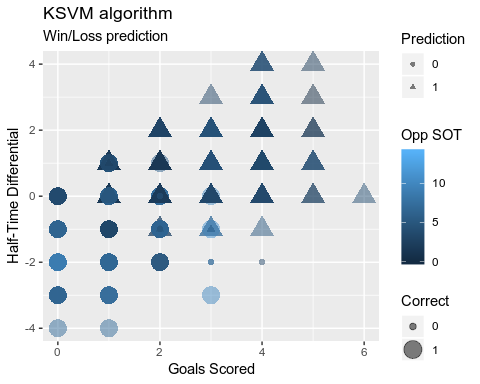
## Warning in data.matrix(newdata): NAs introduced by coercion  
  
## Warning in data.matrix(newdata): NAs introduced by coercion  
  
## Warning in data.matrix(newdata): NAs introduced by coercion  
  
## Warning in data.matrix(newdata): NAs introduced by coercion

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 439 75  
## 1 48 198  
##   
## Accuracy : 0.8382   
## 95% CI : (0.81, 0.8636)  
## No Information Rate : 0.6408   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.6406   
## Mcnemar's Test P-Value : 0.01906   
##   
## Sensitivity : 0.9014   
## Specificity : 0.7253   
## Pos Pred Value : 0.8541   
## Neg Pred Value : 0.8049   
## Prevalence : 0.6408   
## Detection Rate : 0.5776   
## Detection Prevalence : 0.6763   
## Balanced Accuracy : 0.8134   
##   
## 'Positive' Class : 0   
##

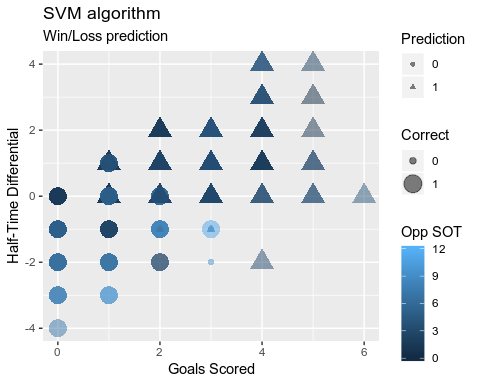
Naive Bayes was closer to KSVM but not close enough. Maybe by further adjusting the models, we can have closer results and maybe even improve to a 90 percent prediction rate.

1. Plotting and further interpreting results. We’ll now plot the three classification algorithms to see how they stack against each other.

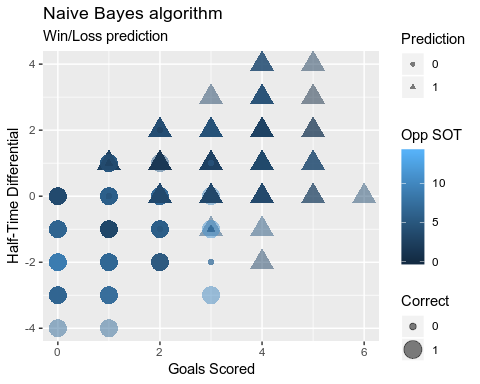
## Warning: Using size for a discrete variable is not advised.



## Warning: Using size for a discrete variable is not advised.



## Warning: Using size for a discrete variable is not advised.



So, we can see that the model correctly predicts most of the matches. Nevertheless, it doesn’t do an overall good job as there are many areas where the model predicts both correctly and incorrectly. This is an area of further improvement within the model that could probably be solved by adding more data points or adjusting the models.

### CONCLUSIONS

1. Final thoughts We managed to answer the three questions that we set out at the start of this project.
2. When viewing the descriptive statistics, there was no significant variation in teams winning home or away - though teams tended to draw more away than at home. We further saw that this was not something worthwhile considering when we dived into the logistic regression. Home-field advantage was thus discarded as something not really present within the English Premier League.
3. On the score at half-time, we did find that teams that entered half-time in the lead ended more likely than not winning the game. However, if both teams entered half-time tied, there was an equal chance of either team winning or both of them tying, which is very interesting. This means that teams are very competitive and any tied game is up for grabs.
4. Finally, we found that fouls committed or received, as well as yellow cards, had no impact on the outcome of the game. Red cards received however where very influential in the result.  
   Similarly, corner kicks taken or allowed, and shots taken had no effect on the end result, but shots on target by the opposing team where a factor that was closely correlated to the result of the game at full-time.

### APPENDIX - RStudio Code

# Load the required packages  
require(arules)  
require(bindr)  
require(caret)  
require(caTools)  
require(dplyr)  
require(e1071)  
require(eeptools)  
require(formatR)  
require(ggplot2)  
require(gridExtra)  
require(kernlab)  
require(lubridate)  
require(magrittr)  
require(neuralnet)  
require(randomForest)  
require(readxl)  
require(reshape2)  
require(rpart)  
require(sqldf)  
require(tidyr)  
require(tidyverse)  
  
# Obtain the data from the appropriate sites.  
prem1516 <- read.csv("http://www.football-data.co.uk/mmz4281/1516/E0.csv")  
prem1617 <- read.csv("http://www.football-data.co.uk/mmz4281/1617/E0.csv")  
prem1718 <- read.csv("http://www.football-data.co.uk/mmz4281/1718/E0.csv")  
  
# Bind the data into one large data frame.  
prem <- prem1516 %>% bind\_rows(prem1617) %>% bind\_rows(prem1718)  
head(prem)  
  
# Transform the dataset into a more comprehensive dataset, with one row for  
# each team and date. Select Home Team and Home team data.  
premHomeTeams <- prem %>% dplyr::mutate(HomeAway = "Home") %>% dplyr::select(Div,   
 Date, Team1 = HomeTeam, Team2 = AwayTeam, HomeAway, GS = FTHG, GA = FTAG,   
 FTR, HTGS = HTHG, HTGA = HTAG, HTR, Referee, ShotsTaken = HS, ShotsAllowed = AS,   
 ShotsOnTarget = HST, OppShotsOnTarget = AST, FoulsCommitted = HF, FoulsReceived = AF,   
 CornerKicks = HC, OppCornerKicks = AC, YellowCards = HY, OppYellowCards = AY,   
 RedCards = HR, OppRedCards = AR, WinOdds = B365H, DrawOdds = B365D, LossOdds = B365A)  
  
# Select Away Team and Away team data.  
premAwayTeams <- prem %>% dplyr::mutate(HomeAway = "Away") %>% dplyr::select(Div,   
 Date, Team1 = AwayTeam, Team2 = HomeTeam, HomeAway, GS = FTAG, GA = FTHG,   
 FTR, HTGS = HTAG, HTGA = HTHG, HTR, Referee, ShotsTaken = AS, ShotsAllowed = HS,   
 ShotsOnTarget = AST, OppShotsOnTarget = HST, FoulsCommitted = AF, FoulsReceived = HF,   
 CornerKicks = AC, OppCornerKicks = HC, YellowCards = AY, OppYellowCards = HY,   
 RedCards = AR, OppRedCards = HR, WinOdds = B365A, DrawOdds = B365D, LossOdds = B365A)  
  
# Bind datasets into a new dataset.  
premTransformed <- premHomeTeams %>% bind\_rows(premAwayTeams)  
  
# Check the first part of the data, column names, dimensions of the dataset,  
# and the structure.  
head(premTransformed)  
colnames(premTransformed)  
dim(premTransformed)  
str(premTransformed)  
  
# Create functions that are able to provide several measurements of central  
# tendency. Calculate mean shots taken for each team.  
performanceGoalsHomeAway <- function(x) {  
 df <- x  
 MeanShotsTaken <- mean(df$ShotsTaken)  
 MeanShotsOnTarget <- mean(df$ShotsOnTarget)  
 MeanCornerKicks <- mean(df$CornerKicks)  
 MeanFTGoals <- mean(df$GS)  
 MeanHTGoals <- mean(df$HTGS)  
   
 mStats <- c(MeanShotsTaken, MeanShotsOnTarget, MeanCornerKicks, MeanFTGoals,   
 MeanHTGoals)  
 return(mStats)  
}  
  
# The function performanceCardsHomeAway computes average of cards received  
# and fouls committed and against Home and Away teams GF: Goal For, GA: Goal  
# Against, AVG  
performanceCardsHomeAway <- function(x) {  
 df <- x  
 TotalGamesPlayed <- nrow(df)  
 TotalFoulsCommitted <- sum(df$HF)  
 TotalFoulsReceived <- sum(df$AF)  
 TotalYellowCards <- sum(df$HY)  
 TotalRedCards <- sum(df$HR)  
 TotalOppYellowCards <- sum(df$AY)  
 TotalOppRedCards <- sum(df$AR)  
 MeanFoulsCommitted <- round(mean(df$HF), digits = 1)  
 MeanFoulsReceived <- round(mean(df$AF), digits = 1)  
 MeanYellowCards <- round(mean(df$HY), digits = 1)  
 MeanRedCards <- round(mean(df$HR), digits = 1)  
 MeanOppYellowCards <- round(mean(df$AY), digits = 1)  
 MeanOppRedCards <- round(mean(df$AR), digits = 1)  
   
 mStats <- data.frame(TotalGamesPlayed, TotalFoulsCommitted, TotalFoulsReceived,   
 TotalYellowCards, TotalRedCards, TotalOppYellowCards, TotalOppRedCards,   
 MeanFoulsCommitted, MeanFoulsReceived, MeanYellowCards, MeanOppYellowCards,   
 MeanRedCards, MeanOppRedCards)  
 return(mStats)  
}  
  
# The function descriptiveStatsInfo’ takes a dataframe as input, and  
# calculates the followings: mean, median, min and max, standard deviation,  
# quantiles, and skewness.  
descriptiveStatsInfo <- function(x) {  
 # Computing statistical measurements for teams  
 df <- x  
 MeanFTWin <- mean(df$TotalFTWin)  
 MeanFTLoss <- mean(df$TotalFTLoss)  
 MeanFTDraw <- mean(df$TotalFTDraw)  
 STDFTWin <- sd(df$TotalFTWin, na.rm = TRUE)  
 STDFTLoss <- sd(df$TotalFTLoss, na.rm = TRUE)  
 STDFTDraw <- sd(df$TotalFTDraw, na.rm = TRUE)  
 MedianFTWin <- median(df$TotalFTWin)  
 MedianFTLoss <- median(df$TotalFTLoss)  
 MedianFTDraw <- median(df$TotalFTDraw)  
 MinFTWin <- min(df$TotalFTWin)  
 MaxFTWin <- max(df$TotalFTWin)  
 MinFTLoss <- min(df$TotalFTLoss)  
 MaxFTLoss <- max(df$TotalFTLoss)  
 MinFTDraw <- min(df$TotalFTDraw)  
 MaxFTDraw <- max(df$TotalFTDraw)  
 QuantilesFTWinInfo <- quantile(df$TotalFTWin, probs = c(0.05, 0.95))  
 SkewnessFTWinInfo <- skewness(df$TotalFTWin)  
 QuantilesFTLossInfo <- quantile(df$TotalFTLoss, probs = c(0.05, 0.95))  
 SkewnessFTLossInfo <- skewness(df$TotalFTLoss)  
 QuantilesFTDrawInfo <- quantile(df$TotalFTDraw, probs = c(0.05, 0.95))  
 SkewnessFTDrawInfo <- skewness(df$TotalFTDraw)  
   
 mStats <- data.frame(MeanFTWin, MeanFTLoss, MeanFTDraw, STDFTWin, STDFTLoss,   
 STDFTDraw, MedianFTWin, MedianFTLoss, MedianFTDraw, MinFTWin, MaxFTWin,   
 MinFTLoss, MaxFTLoss, MinFTDraw, MaxFTDraw, QuantilesFTWinInfo, SkewnessFTWinInfo,   
 QuantilesFTLossInfo, SkewnessFTLossInfo, QuantilesFTDrawInfo, SkewnessFTDrawInfo)  
   
 return(mStats)  
}  
  
# The function summaryStatsHomeAway computes totals for Home and Away teams  
summaryStatsHomeAway <- function(x, mTeam1, mHomeAway) {  
 df <- x  
 HomeAway <- mHomeAway  
 Team1 <- mTeam1  
 TotalGamesPlayed <- nrow(df)  
 TotalShotsTaken <- sum(df$ShotsTaken)  
 TotalShotsOnTarget <- sum(df$ShotsOnTarget)  
 TotalFoulsCommitted <- sum(df$FoulsCommitted)  
 TotalCornerKicks <- sum(df$CornerKicks)  
 TotalYellowCards <- sum(df$YellowCards)  
 TotalRedCards <- sum(df$RedCards)  
 TotalFTGoals <- sum(df$GS)  
 TotalHTGoals <- sum(df$HTGS)  
 df$FTR <- gsub(" ", "", df$FTR)  
 df$HTR <- gsub(" ", "", df$HTR)  
 TotalHTWin <- length(which(df$HTR == "H"))  
 TotalHTLoss <- length(which(df$HTR == "A"))  
 TotalHTDraw <- length(which(df$HTR == "D"))  
 TotalFTWin <- length(which(df$FTR == "H"))  
 TotalFTLoss <- length(which(df$FTR == "A"))  
 TotalFTDraw <- length(which(df$FTR == "D"))  
 mStats <- data.frame(Team1, HomeAway, TotalShotsTaken, TotalShotsOnTarget,   
 TotalFoulsCommitted, TotalCornerKicks, TotalYellowCards, TotalRedCards,   
 TotalFTGoals, TotalHTGoals, TotalHTWin, TotalHTLoss, TotalHTDraw, TotalFTWin,   
 TotalFTLoss, TotalFTDraw, TotalGamesPlayed)  
 return(mStats)  
}  
  
# The function performanceHomeAdvantage computes the Home Field Advantage  
performanceHomeAdvantage <- function(x) {  
 # Home field advantage will be calculated as follows: The team gets 3 points  
 # for a win, 1 point for a draw, and zero for a loss. Calculate the ratio  
 # of total points earned from all matches/games played at home vs. away. If  
 # the ratio is 50%, it indicates that success has been achieved equally at  
 # home and away i.e.there is no home field advantage. If the ratio is above  
 # 50% then it indicates that playing at home field increases the winning  
 # probability.  
 df <- x  
 mMax <- nrow(df)  
 mCounter <- 1  
 HomeAdvantagedf <- data.frame()  
 while (mCounter <= (mMax - 1)) {  
 TotalHomePoints <- 0  
 TotalAwayPoints <- 0  
 if (df$HomeAway[mCounter] == "Home") {  
 TotalHomePoints <- df$TotalFTWin[mCounter] \* 3 + df$TotalFTDraw[mCounter] \*   
 1  
 TotalAwayPoints <- df$TotalFTWin[mCounter + 1] \* 3 + df$TotalFTDraw[mCounter] \*   
 1  
 }  
 HomeAdvantage = round(TotalHomePoints/(df$TotalGamesPlayed[mCounter] \*   
 3) \* 100, digits = 1)  
 AwayAdvantage = round(TotalAwayPoints/(df$TotalGamesPlayed[mCounter] \*   
 3) \* 100, digits = 1)  
   
 mStats <- data.frame(df$Team1[mCounter], HomeAdvantage, AwayAdvantage,   
 df$TotalGamesPlayed[mCounter])  
 HomeAdvantagedf <- HomeAdvantagedf %>% rbind(mStats)  
   
 mCounter <- mCounter + 2  
 }  
 return(HomeAdvantagedf)  
}  
  
# Step 2: Create a dataframe with all teams (Home and Away) participating in  
# the UK premiere league and their totals.  
premTeams <- gsub(" ", "", premTransformed$Team1)  
premTeams <- unique(premTeams)  
premTeams  
  
premHome <- premHomeTeams %>% mutate(HomeAway = "Home") %>% select(Date, Team1,   
 Team2, HomeAway, GS, FTR, HTGS, HTR, ShotsTaken, ShotsOnTarget, FoulsCommitted,   
 CornerKicks, YellowCards, RedCards)  
premAway <- premAwayTeams %>% mutate(HomeAway = "Away") %>% select(Date, Team1,   
 Team2, HomeAway, GS, FTR, HTGS, HTR, ShotsTaken, ShotsOnTarget, FoulsCommitted,   
 CornerKicks, YellowCards, RedCards)  
  
mCounter <- 1  
mMax <- length(premTeams)  
premHomeTeamStat <- data.frame()  
premAwayTeamStat <- data.frame()  
premStat <- data.frame()  
while (mCounter <= mMax) {  
 mTeam <- premTeams[mCounter]  
 premHomeTeam <- premHome[which(premHome$Team1 == mTeam), ]  
 premAwayTeam <- premAway[which(premAway$Team1 == mTeam), ]  
 premHomeStatTemp <- summaryStatsHomeAway(premHomeTeam, mTeam, "Home")  
 premAwayStatTemp <- summaryStatsHomeAway(premAwayTeam, mTeam, "Away")  
 premHomeTeamStat <- cbind(premHomeStatTemp)  
 premAwayTeamStat <- cbind(premAwayStatTemp)  
 premStat <- premStat %>% rbind(premHomeTeamStat)  
 premStat <- premStat %>% rbind(premAwayTeamStat)  
 mCounter <- mCounter + 1  
}  
  
# Renumber/reindex the row numbers  
row.names(premStat) <- NULL  
# premStat <- premStat[,-2:-2] # removing Team2  
premStat  
  
# Step 3: calculate the descriptive & performance statistics for all Home  
# and Away teams.  
  
# 1. Calculate average goals scorded for or against for Home and Away teams  
premHomeTeamGoals <- performanceGoalsHomeAway(premHome)  
premAwayTeamGoals <- performanceGoalsHomeAway(premAway)  
premHomeTeamGoals  
premAwayTeamGoals  
  
# 2. Calculate average cards received and fouls committed for Home and Away  
# teams  
premTeamCards <- performanceCardsHomeAway(prem)  
premTeamCards  
  
# 3. Calculate descriptive statistics for Home and Away teams  
premHomedescriptiveStats <- data.frame()  
premAwaydescriptiveStats <- data.frame()  
premdescriptiveStats <- data.frame()  
premHomedescriptiveStats <- descriptiveStatsInfo(premStat[which(premStat$HomeAway ==   
 "Home"), ])  
premAwaydescriptiveStats <- descriptiveStatsInfo(premStat[which(premStat$HomeAway ==   
 "Away"), ])  
premdescriptiveStats <- premHomedescriptiveStats %>% rbind(premAwaydescriptiveStats)  
row.names(premdescriptiveStats) <- c("Home05", "Home95", "Away05", "Away95")  
premdescriptiveStats <- descriptiveStatsInfo(premStat)  
str(premdescriptiveStats)  
View(premdescriptiveStats)  
  
# Calculate mean wins, draws, and losses for each team and create plots.  
dfpremStat <- data\_frame()  
dfpremStat <- premStat[which(premStat$HomeAway == "Home"), ]  
dfpremStat$Team1 <- gsub(" ", "", dfpremStat$Team1)  
dfpremStat <- dfpremStat[order(dfpremStat$Team1), ]  
row.names(dfpremStat) <- NULL  
dfpremStat <- dfpremStat[-2, ]  
dfpremStat <- dfpremStat[-9, ]  
dfpremStat <- dfpremStat[-13, ]  
dfpremStat <- dfpremStat[-14, ]  
dfpremStat <- dfpremStat[-16, ]  
row.names(dfpremStat) <- NULL  
TeamCodes <- c("ARS", "BOU", "BRH", "BUR", "CHE", "CRY", "EVE", "HDD", "LEI",   
 "LIV", "MCI", "MUN", "NEW", "SOU", "STK", "SWA", "TOT", "WAT", "WBA", "WHU")  
dfpremStat <- dfpremStat %>% cbind(TeamCodes)  
dfTotalWin <- cbind(dfpremStat$TeamCodes) %>% cbind(dfpremStat$TotalFTWin)  
dfpremStat <- dfpremStat[, -1:-13]  
dfpremStat <- dfpremStat[, -4]  
  
# Full-time wins  
ggplot(dfpremStat, aes(x = dfpremStat$TotalFTWin)) + geom\_histogram(aes(y = ..density..),   
 binwidth = 3, colour = "white", fill = "red") + geom\_density(alpha = 0.2,   
 fill = "blue") + labs(title = "English Premier League - Distribution of Total Wins",   
 x = "Total Wins") + theme\_bw() + theme(axis.ticks = element\_blank(), axis.text.y = element\_blank())  
  
# Full-time losses  
ggplot(dfpremStat, aes(x = dfpremStat$TotalFTLoss)) + geom\_histogram(aes(y = ..density..),   
 binwidth = 3, colour = "white", fill = "red") + geom\_density(alpha = 0.2,   
 fill = "blue") + labs(title = "English Premier League - Distribution of Total Losses",   
 x = "Total Losses") + theme\_bw() + theme(axis.ticks = element\_blank(), axis.text.y = element\_blank())  
  
# Full-time draws  
ggplot(dfpremStat, aes(x = dfpremStat$TotalFTDraw)) + geom\_histogram(aes(y = ..density..),   
 binwidth = 3, colour = "white", fill = "red") + geom\_density(alpha = 0.2,   
 fill = "blue") + labs(title = "English Premier League - Distribution of Total Draws",   
 x = "Total Draws") + theme\_bw() + theme(axis.ticks = element\_blank(), axis.text.y = element\_blank())  
  
# Calculate mean measurements for disciplinary statistics (fouls, yellow and  
# red cards.)  
premTeamCards <- performanceCardsHomeAway(prem)  
premCards <- data.frame()  
premCards <- premCards %>% rbind(premTeamCards)  
premCards <- premCards %>% rbind(premCards)  
colnames(premCards) <- c("TotalGamesPlayed", "TotalFoulsCommitted", "TotalFoulsReceived",   
 "TotalYellowCards", "TotalRedCards", "TotalOppYellowCards", "TotalOppRedCards",   
 "MeanFoulsCommitted", "MeanFoulsReceived", "MeanYellowCards", "MeanOppYellowCards",   
 "MeanRedCards", "MeanOppRedCards")  
row.names(premCards) <- c("Cards Received", "Cards Opponent")  
mTotalGamesPlayed <- premTeamCards[1]  
premCards <- premCards[, 4:7]  
premCards[2, 1] <- premCards[1, 3]  
premCards[2, 2] <- premCards[1, 4]  
premCards <- premCards[, 1:2]  
premCardsTemp <- premCards$TotalYellowCards  
premCardsTemp <- premCardsTemp %>% cbind(premCards$TotalRedCards)  
  
mylabels <- c("Yellow Cards", "Yellow Cards", "Red Cards", "Red Cards")  
barplot(premCardsTemp, main = paste("Total Cards Received in", mTotalGamesPlayed,   
 "Games"), ylab = "Number of Cards", xlab = "Card Type", names.arg = mylabels,   
 ylim = c(0, 2000), col = c("red", "blue"), beside = TRUE)  
legend("right", c("Home", "Away"), fill = c("red", "blue"), cex = 0.75)  
  
# Check if there is a difference in scoring and shots taken.  
premHomeTeamGoals <- performanceGoalsHomeAway(premHome)  
premAwayTeamGoals <- performanceGoalsHomeAway(premAway)  
premTeamGoals <- data\_frame()  
premTeamGoals <- rbind(premHomeTeamGoals) %>% rbind(premAwayTeamGoals)  
colnames(premTeamGoals) <- c("Shots Taken", "Shots On Target", "CornerKicks",   
 "Full Time Goals", "Half Time Goals")  
mylabels <- c("Shots Taken", "Shots On Target", "Corner Kicks", "Full Time Goals",   
 "Half Time Goals")  
barplot(premTeamGoals, main = "Average Shots & Goals Scorded by Home & Away Teams",   
 xlab = "Shots & Goals", ylab = "Mean Score", ylim = c(0, 15), names.arg = mylabels,   
 col = c("#E82C0C", "#340CB5"), beside = TRUE)  
legend("topright", c("Home", "Away"), fill = c("#E82C0C", "#340CB5"))  
  
# Finally, check if there's a significant home-field advantage.  
premHomeAdvantage <- data.frame()  
premStat <- premStat[order(premStat$Team1, premStat$HomeAway), ]  
dfHomeAdvantage <- performanceHomeAdvantage(premStat)  
  
colnames(dfHomeAdvantage) <- c("Team", "Home Advantage", "Away Advantage", "Total Games Played")  
dfHomeAdvantage$Team <- gsub(" ", "", dfHomeAdvantage$Team)  
dfHomeAdvantage <- dfHomeAdvantage[order(dfHomeAdvantage$Team), ]  
row.names(dfHomeAdvantage) <- NULL  
TeamCodes <- c("ARS", "BOU", "BRH", "BUR", "CHE", "CRY", "EVE", "HDD", "LEI",   
 "LIV", "MCI", "MUN", "NEW", "SOU", "STK", "SWA", "TOT", "WAT", "WBA", "WHU")  
dfHomeAdvantage <- dfHomeAdvantage[-2, ]  
dfHomeAdvantage <- dfHomeAdvantage[-9, ]  
dfHomeAdvantage <- dfHomeAdvantage[-13, ]  
dfHomeAdvantage <- dfHomeAdvantage[-14, ]  
dfHomeAdvantage <- dfHomeAdvantage[-16, ]  
dfHomeAdvantage <- dfHomeAdvantage[, 2:3]  
  
dfHomeAdvantage <- dfHomeAdvantage %>% cbind(TeamCodes)  
  
# Reshape data into long format  
TeamCodes <- dfHomeAdvantage$TeamCodes  
dfHomeAdvantage <- melt(dfHomeAdvantage, id = c("TeamCodes"))  
  
# Build the plot  
ggplot(dfHomeAdvantage) + geom\_bar(aes(x = TeamCodes, y = value, fill = variable),   
 stat = "identity", position = "dodge", width = 0.7) + scale\_fill\_manual("Advantage\n",   
 values = c("red", "blue"), labels = c("Home", "Away")) + labs(title = "English Premier League Teams (Home vs. Away Advantage)",   
 x = "Teams", y = "Percent") + theme\_bw(base\_size = 14) + theme(axis.ticks = element\_blank(),   
 axis.text.y = element\_blank())  
  
# Data modeling. Create a win dummy variable.  
premTransformed$win <- ifelse(premTransformed$GS > premTransformed$GA, 1, 0)  
# Note: a tie is categorized as a 0 in this formula and prediction  
  
# Calculate the score difference at half-time.  
premTransformed$HTdiff <- premTransformed$HTGS - premTransformed$HTGA  
  
# Calculate the sum of red and yellow cards accumulated by both teams.  
premTransformed$red\_yellow <- premTransformed$RedCards + premTransformed$YellowCards  
premTransformed$opp\_red\_yellow <- premTransformed$OppRedCards + premTransformed$OppYellowCards  
  
# Transform HomeAway into a factor.  
premTransformed$HomeAway <- as.factor(premTransformed$HomeAway)  
  
# Generate our first logistic regression taking into account all the  
# variables.  
logModel <- glm(formula = win ~ HomeAway + FTR + HTdiff + HTR + ShotsTaken +   
 ShotsAllowed + ShotsOnTarget + OppShotsOnTarget + FoulsCommitted + FoulsReceived +   
 CornerKicks + OppCornerKicks + YellowCards + OppYellowCards + RedCards +   
 OppRedCards + DrawOdds + LossOdds + red\_yellow + opp\_red\_yellow, data = premTransformed,   
 family = binomial(logit))  
summary(logModel)  
  
# Second logistic regression with most significant variables.  
logModelCondensed <- glm(win ~ HTdiff + ShotsOnTarget + OppShotsOnTarget + CornerKicks +   
 OppCornerKicks + RedCards + OppRedCards, data = premTransformed, family = binomial(logit))  
  
summary(logModelCondensed)  
  
# Final logistic regression with the most significant variables from the  
# second iteration.  
logModelCondensed2 <- glm(win ~ GS + HTdiff + OppShotsOnTarget + RedCards, data = premTransformed,   
 family = binomial(logit))  
  
summary(logModelCondensed2)  
  
# Separate the data into training and testing sets.  
randIndex <- sample(1:dim(premTransformed)[1])  
# selects the cut off point for 67 percent of the data.  
cutPoint2\_3 <- floor(2 \* dim(premTransformed)[1]/3)  
  
# creates training dataset.  
trainData <- premTransformed[randIndex[1:cutPoint2\_3], ]  
# creates test dataset.  
testData <- premTransformed[randIndex[(cutPoint2\_3 + 1):dim(premTransformed)[1]],   
 ]  
  
# Convert the response variable to factor.  
trainData$win <- as.factor(trainData$win)  
testData$win <- as.factor(testData$win)  
  
# Generate a KSVM classification model.  
ksvmPrem <- ksvm(win ~ GS + HTdiff + OppShotsOnTarget + RedCards, data = trainData,   
 kernel = "rbfdot", kpar = "automatic", C = 10, cross = 10, prob.model = TRUE)  
  
# Predict and transform the data to test accuracy  
ksvmPred <- predict(ksvmPrem, testData)  
ksvmPred <- as.data.frame(ksvmPred)  
colnames(ksvmPred) <- "predicted"  
ksvmPredDf <- testData %>% select(actual = win) %>% bind\_cols(ksvmPred)  
  
# Confusion matrix  
confusionMatrix(ksvmPredDf$predicted, ksvmPredDf$actual)  
  
# Test a SVM classification algorithm.  
svmPrem <- svm(win ~ GS + HTdiff + OppShotsOnTarget + RedCards, data = trainData,   
 kernel = "radial", C = 10, cross = 10, prob.model = TRUE)  
  
# Predict the accuracy of the SVM model  
svmPred <- predict(svmPrem, testData)  
svmPred <- data.frame(svmPred)  
colnames(svmPred) <- "predicted"  
svmPredDf <- merge(testData, svmPred, by = 0, all = TRUE)  
svmPredDf <- svmPredDf %>% filter(!is.na(predicted))  
  
# Build the confusion matrix  
confusionMatrix(svmPredDf$predicted, svmPredDf$win)  
  
# Build a Naive Bayes algorithm.  
nbPrem <- naiveBayes(win ~ GS + HTdiff + OppShotsOnTarget + RedCards, data = trainData)  
  
# Predict, transform, and build the confusion matrix.  
nbPred <- predict(nbPrem, testData)  
nbPred <- data.frame(nbPred)  
colnames(nbPred) <- "predicted"  
nbPredDf <- testData %>% select(actual = win) %>% bind\_cols(nbPred)  
  
confusionMatrix(nbPredDf$predicted, nbPredDf$actual)  
  
# Plot the results of the different models. Plot KSVM model  
ksvmPredDf <- testData %>% bind\_cols(ksvmPred) %>% mutate(correct = ifelse(win ==   
 predicted, 1, 0))  
  
ksvmPlot <- ggplot(ksvmPredDf, aes(x = GS, y = HTdiff, color = OppShotsOnTarget)) +   
 geom\_point(aes(size = as.factor(correct), shape = as.factor(predicted)),   
 alpha = 0.5) + ggtitle("KSVM algorithm", subtitle = "Win/Loss prediction") +   
 labs(x = "Goals Scored", y = "Half-Time Differential", size = "Correct",   
 color = "Opp SOT", shape = "Prediction")  
  
# Plot SVM Model  
svmPredDf <- svmPredDf %>% mutate(correct = ifelse(win == predicted, 1, 0))  
  
svmPlot <- ggplot(svmPredDf, aes(x = GS, y = HTdiff, color = OppShotsOnTarget)) +   
 geom\_point(aes(size = as.factor(correct), shape = as.factor(predicted)),   
 alpha = 0.5) + ggtitle("SVM algorithm", subtitle = "Win/Loss prediction") +   
 labs(x = "Goals Scored", y = "Half-Time Differential", size = "Correct",   
 color = "Opp SOT", shape = "Prediction")  
  
# Plot Naive Bayes Model  
nbPredDf <- testData %>% bind\_cols(nbPred) %>% mutate(correct = ifelse(win ==   
 predicted, 1, 0))  
  
nbPlot <- ggplot(nbPredDf, aes(x = GS, y = HTdiff, color = OppShotsOnTarget)) +   
 geom\_point(aes(size = as.factor(correct), shape = as.factor(predicted)),   
 alpha = 0.5) + ggtitle("Naive Bayes algorithm", subtitle = "Win/Loss prediction") +   
 labs(x = "Goals Scored", y = "Half-Time Differential", size = "Correct",   
 color = "Opp SOT", shape = "Prediction")  
  
# Print the plots  
ksvmPlot  
svmPlot  
nbPlot