Modelling Football Data

Ralph Braithwaite
Mathematics and Statistics
Coventry University

Contents

1 Introduction

In this project we hope to examine generalised linear regression models and see how they can be applied to data from the English Premier League.
We will first look at logistic regression models, to see how match result is affected by a few different effects. Logistic regression can be used to model Bernoulli processes with variable probability dependent on different factors, making it a useful tool for predicting outcome in sports.
Afterwards we will look at proportional hazards models and see how they can be used for analysis of first-goal times. Proportional hazards models that we hope to focus on are parametric models of exponential or Weibull type, and the Cox proportional-hazards model.

Notes

Word count excluding bibliography, appendices is approx 11.7k.

This document was typeset using Typst. The code used for generating page headers was written by Jacopo Zagoli, and used under the terms of the Apache open source license. Their original work and the associated license can be found at the following link. https://github.com/zagoli/simple-typst-thesis

All code pertaining to statistics was written in R, though some helper scripts were written in Python. All relevant code will be in the appendices, and some time after the Turnitin check all source files will be uploaded to my Github: https://github.com/ralphmb

This project received ethics approval from Coventry University, the certificate can be found at the end of the appendices.

Thank you to Prof. Houshang Mashhoudy for his practical support and guidance, and thank you to Deimantė Bogužaitė for her moral support and cooking.

2 Review of Literature

There is a wealth of literature concerning itself with the statistical modelling of football. For fans of the sport it is perhaps the unpredictability of the game that makes it appealing, as while team strength is of course a large factor in game outcome, the multitude of other causal variables can lead to upset wins and tightly-fought battles. This same reasoning might explain the large base of statistical research into the game, as statisticians and sport scientists seek to quantify the effects of myriad different factors. Football is big business, with the English Premier League (EPL) having contributed £7.6 billion to the UK's gross domestic product in 2019/2020 [1], invested throughout the country and disproportionately in more deprived areas such as the north west. In the year preceding Dec 2020, 2.5 million Britons placed bets on Football [2], and the Financial Times notes that in the year to March 2022 £1.1bn was taken in revenue for gambling operators, nearly half the £2.3bn figure for their total revenue [3]. Statistical prediction of football matches is naturally very important to bookmakers, seeking to maximise their revenues by setting optimal odds for games.

Linear models predicting match outcome based on various pre-match variables are ubiquitous, but there are examples of authors applying time-series techniques to assess minute-to-minute statistics based on in-game events such as corner kicks and yellow cards. Some authors have applied Bayesian network techniques [4] and in many more recent papers we see the use of machine learning models to answer similar questions. In this section we hope to explore a limited number of techniques applied to football modelling.

2.1 Logistic Regression

Logistic regression is a common tool for modelling situations where outcomes can be classified as a binary variable. In football, the result of a match is naturally three-valued - win vs draw vs loss - but this can be recast as a binary response: win vs not a win. It is used as a tool for predicting outcome in many sports [5], [6], but its use in football is well trodden. Here we will study one example. D. Prasetio and D. Harlili [7] train a logistic regression model on variables representing defense/offense ratings against the logit of win chance.

The authors use win/loss data from the English premier league from between 2010 and 2016. They fit a number of models, excluding different subsets of the data for each, in order to examine to predictive usefulness of older data and for out-of-sample forecasting of newer data. All the models are tested on 2015/16 season data. The authors determine that the defense variables are more significant than the offense ones, and examine the tradeoff between losing some predictive power when the offense variables are dropped, vs having a simpler model. Part of their conclusion is that the ease of interpretation of logistic regression makes it a useful tool, as well as the good power of the technique to make accurate predictions.

2.2 Survival Analysis

Goals are the main event of interest for both fans and players in football, and the time-to-first-goal (TTFG) is a key measure for research. Intuitively this may be seen as a measure of a team's offensive prowess over their opponents, and indeed studies have shown it to be a very useful predictor of match outcome: S. Ibáñez, J. Pérez-Goye, J. Courel Ibáñez, and J. García [8] conducted a study of European women's football and found that teams who score first were 5 times more likely to win. Survival analysis is a natural tool for modelling this, as it concerns itself with durations until or between events that may not be fully observed. In football terms this is because the TTFG is often cut off, or censored, by the end of the match. Some authors censor at other events, for instance at points when the other team scores, as this changes the dynamic of the game. Modelling censored time-to-event data can be done in many ways. Cox regression is a common tool, particularly in medical statistics, though parametric models are also common.

A paper by D. Nevo and Y. Ritov [9] undertakes an in-depth analysis of the times to first and second goals using the Cox proportional hazard regression model. Their analysis covers around 760 matches between 2008 and 2010 in the English premier league. They use data on the times in minutes of the first and second goals, as well as some other information such as the times that red cards are shown to a team, as well as the season and home team.

Goal times are censored at 90 minutes if they do not occur during the match, and time-to-second-goal is left-truncated since it cannot occur until the first goal has been scored. Their given reason for focusing on the first two goals is that they'd find much heavier censoring if they included data for third or fourth goals - as football tends to be a fairly low-scoring sport, this would then require far more data to make a model accurate at higher goal counts. Two nested models for

the hazard function are introduced, a null model and a complete one. The complete Cox hazard function is given as follows:

$$
\begin{aligned}
h_{ij}(t) = h_0(t)\exp\big(&\\
&\beta_1 \text{ProbWin}_i + \beta_2 \text{Season}_i\\
&+\beta_3 \text{RedCardsAway}_i(t)\\
&+I_{j=2}(\beta_4 \text{Goal}_i\\
&\quad +\beta_5 \text{TimeOfFirstGoal}_i\\
&\quad +\beta_6 \text{FirstGoalTeam}_i\\
&\quad +\beta_7 \text{TimeFromFirstGoal}_i)\\
\big)&
\end{aligned}
$$

Where $j \in \{1, 2\}$ is the goal number, $i \in \{1, ..., 760\}$ is the match number, and $I_{j=2}$ is an indicator function. The function I has the purpose of changing the hazard when the first goal has been scored. All variables here except FirstGoalTeam are kept, though the significance of Goal and TimeFromFirstGoal are low ($p = 0.058, 0.11$ resp). The authors experiment with a 'frailty' term to account for random effects, however this is later shown to not be significant. They find that red cards for the away team raise the hazard by 1.806 times, and that the change in hazard after the first goal depends on the time of the match, going down if the first goal occurs before 51 minutes and up afterwards. The authors draft some ideas to explain parts of the model, for instance that the insignificance of FirstGoalTeam may be due to tactical reasons, where after a goal a team may become more aggressive and both teams will find it easier to score further goals. The baseline cumulative hazard function $\hat{H}_0(t)$ seems linear, implying an exponential distribution for goal time. A similar model is fit to the same data for the away team, so censoring works the opposite way. Some conclusions drawn from the away model include evidence toward away teams playing more conservatively, likely due to the away disadvantage that we discuss later.

2.3 Weibull Models

A fairly natural way to model the number of goals scored in a match is as a Poisson process. Some authors [10] model the goals of each team as separate Poisson processes, others [11] model the total goals scored as Poisson, assigning each goal to a team according to a binomial distribution. An oft-cited 2019 paper by T. Kharrat and G. N. Boshnakov [12] highlights an issue with the Poisson approach - it begins by stating that:

"... the main issue with the Poisson model when modelling the goals scored by a team in football is that the hazard function [...] remains constant for every time unit [...] However, empirical studies showed that this is rather questionable..."

Kharrat and Boshnakov explore an alternate class of models, based on the Weibull distribution. The Weibull distribution is a generalisation of the exponential distribution, as it adds a shape parameter k that allows for a variable hazard rate. Modelling the incidence rate of goals scored as following a Weibull distribution leads to a Weibull-count model, which generalises the Poisson model. When $k = 1$ the two models are equivalent. Using the R statistical programming language they fit two seperate models to a publically available set of football data. The first is a general linear model with goals scored based on the Poisson distribution, and the second is a renewal-count model based on the Weibull distribution. In a visual analysis of the actual goal counts compared to the counts predicted by each model, the Weibull model appears to more closely match reality, this is later confirmed formally. Due to the aforementioned property of these models that one nests the other, the authors can use a likelihood ratio test (LRT) to compare them. They find that to a $p < 0.001$ confidence level that the additional fit parameter introduced by the Weibull model does improve the fit, and conclude that the Poisson model is outclassed for modelling this set of data.

2.4 Home Advantage

Home advantage (HA) has a major effect on match outcome in football, and in fact research [13] has shown that across 10 different sports the HA effect is significantly highest in football. Various authors try to relate HA to the distance that athletes must travel, the pressure from crowds of local fans on the players and the referee, or other psychological effects on the players from being in an unfamiliar environment.

R. Pollard and V. Armatas [14] discusses the factors that determine home advantage in World Cup qualifying matches. The authors examine 2040 games played by national teams during the 2006, '10 and '14 seasons. They analyse the effect of home advantage on qualification points. While the 'strength' of a team is not directly measurable, for the purpose of quantifying it the authors use FIFA rankings.

Other variables examined include details about match attendance, distance of the crowd from play, number of time zones the away team crossed to attend, altitude change, referee bias and the continental re-

gion in which the teams competed. Variables spanning multiple orders of magnitude underwent log-transformations, and multicollinearity between the large number of variables was checked and found to be negligible. The authors fit a generalised linear model (GLM) for qualification points against these data. This GLM had $R^2 = 0.326$, suggesting that the chosen variables are significant predictors of game points. The most significant variable was found to be the team's ranking, a representation of its strength. Home advantage was found to be significantly higher in Africa than in Europe or Asia, but the differences between other continental areas were found to be insignificant. The authors also find that altitude change and time zones crossed for the away team are both significant effects, as well as the number of attendees. Their results indicate that neither attendance as a percentage of capacity nor distance travelled by the away team had a significant effect ($p = 90\%, 6.7\%$ respectively). This last result was perhaps surprising as other authors [15] have found that distance travelled is a significant factor.

3 Preliminary Steps

3.1 Data Collection

The bulk of our data were obtained from FootyStats for a fee, selecting the 2022/23 season option and downloading the .csv called Match H2H CSV. Data on the locations of each team's stadium were compiled ourselves using google maps. Data on the standings and scores of each team from the previous EPL season were obtained from TNTSports, who allow for non-commercial use.

We will use information on the distances that an away team has to travel to attend a match. These distances were computed using the co-ordinate data mentioned above, and the `distVincentyEllipsoid` function in the `geosphere` package in R. We could have taken travel distance data directly from google maps, which would also calculate actual distance rather than as-the-crow-flies, but this would require either manually collecting 190 distances or automation using the API.

The previous seasons standings data were copied into a spreadsheet and cleaned up, changing team names to follow the same standard as the FootyStats data, before being exported to a .csv file.

The relevant data from FootyStats were extracted, some processed, and saved in a new file. Processing included reading the first goal time out of a string containing a list of all goal times, for instance taking the string "10,27,64" into the numeric value 10.

For a 3rd party source on what exactly to count as a derby game we used an article by The Mirror [16]. This was processed using a python script (see appendices) and included in the data preparation R script.

We will use data on the number of league points attained by a team in the previous Premier League season, for a short discussion on why we use this variable in particular please see the 'Testing correlations' section under exploratory analysis.

3.2 Exploratory Analysis

In this section we will examine our data set, to get some idea of the magnitude of various effects on match result.

3.2.1 Testing correlations between variables

We considered various variables as proxies for team strength. We wanted to use a measure that exists before the beginning of a league season, so results in the previous league seemed obvious. Data on the number of wins in the previous league, total goal difference, and league points were assembled. As is to be expected, these are highly correlated, with pairwise Pearson correlations between 0.94 and 0.98. Since each variable almost entirely explains the others, from here on we will only use league points. Total points seems most relevant to the teams as it is the foremost factor in determining the team's league ranking, and thus the team's prospects of winning or of relegation. For applications requiring factor variables (e.g. Kaplan Meier curves) we will use a variable called `grouping`, a factor variable encoding whether or not the team scored above the median number of points. Teams promoted from a lower league, who have no EPL league points from last season, will be assigned to the lower group.

We also wanted to test for a link between strength and location, theorising that perhaps teams in the center of the country and London would have higher local population density, from which they could draw more support in terms of revenue and choice in young players. To do this we calculated a Pearson rank statistic for the median distance a team had to travel for their away games, against their last-season points. This confirmed no significant link between the two ($\rho = 0.14, t_{\text{df}=15} = 0.58, p = 57\%$), hence we can discount this as a confounding effect.

3.2.2 Home results vs away

Figure 1 shows a bar chart of match results, split by whether the team was playing at home or away.

Home teams won 49.2% of the matches, with 22.3% of matches a tie, and 28.3% were won by the away side. The advantage enjoyed by teams playing at home is clearly visible, and is about on par with results from other recent EPL seasons.

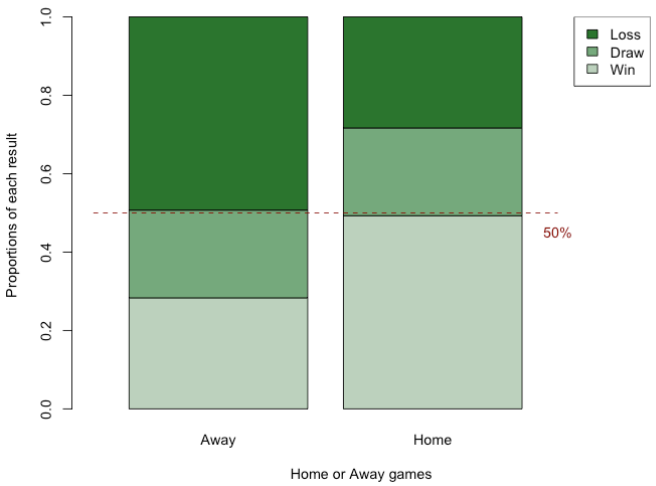


Figure 1: Match results for teams playing at home vs away.

3.2.3 Match results by team strength

We can ignore the effects of home advantage by looking at the results of each match for each team, both the home and away side.

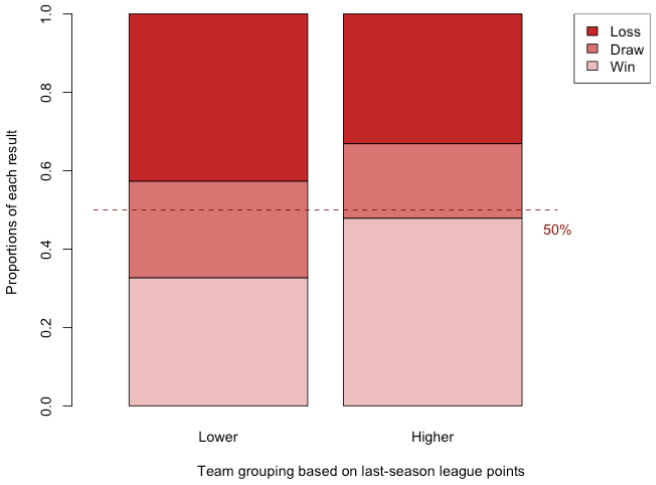


Figure 2: Match results for lower and higher grouped teams

Figure 2 takes these side-agnostic results and splits them by whether the team attained above or below median league points in the previous season. We use this as a proxy for looking at stronger vs weaker teams. We can see that higher-grouped teams, naturally, had a higher chance of winning at 47.8%, so slightly less than average win rate for home teams. The chance of both ties and losses are smaller than among lower-ranked teams (19.0% vs 24.6%, 33.1% vs 42.6% resp.).

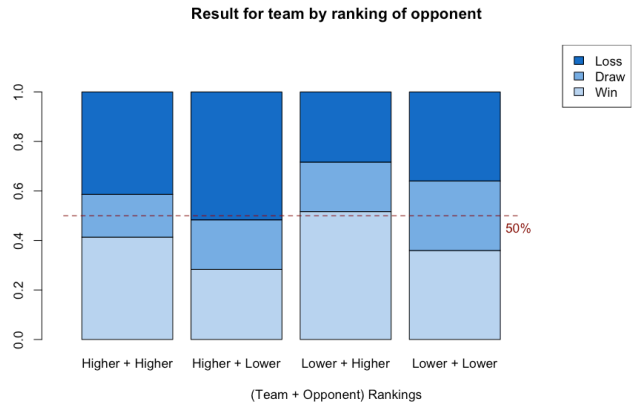


Figure 3: Match results by groupings of either side

For more fine-grained information on the distribution of match results by grouping, we can plot against the grouping of each side. From Figure 3 we can see that games among two higher-grouped teams tend to be more decisive than among two lower-grouped teams. as the chance of a draw on the farthest right bar is 17.3%, much lower than 28.1% on the left. Higher grouped teams can be seen to win 51.7% of the time against those in the lower group, a win chance higher even than for home teams vs away. In these unbalanced games we can see the rate of draws, 20%, is between the rates in games with equally grouped teams. This could suggest that team strength is a stronger determiner of outcome at higher levels, with results being more unpredictable among if one or more teams is on the weaker end.

3.2.4 Interaction between side and strength

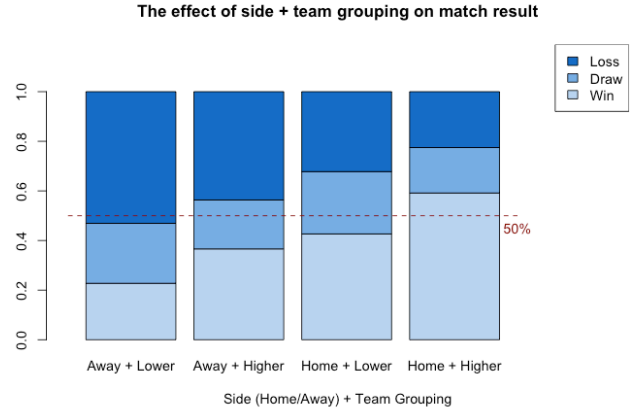


Figure 4: Match results for teams by home vs away and grouping

Figure 4 shows the results for each side based on their grouping and whether they were playing at

home or away. We see again that draw rate (left-to-right: 24.2%, 19.7%, 25.1%, 18.3%) is increased among matches with a lower-grouped team. Higher ranked teams playing at home win 59.1% of their games, 22.5% higher than their win rate away. This difference in win rate is larger than among lower ranked teams, who enjoy only a $42.7\% - 22.8\% = 19.9\%$ higher win rate at home. Similarly, higher grouped teams playing away have a 13.9% higher win rate than lower grouped teams away, below the 16.5% difference among differently grouped home teams. This stacked effect could suggest an interaction between team strength and game side on win rate.

3.2.5 Win Margin

Across every game in the season, the average win margin (home goals − away goals) was 0.48, so in a sense the home advantage can be quantified as being about equivalent to a half-goal headstart. Perhaps just coincidence, but this effect seems to be fairly consistent across time and league, as it's remarkably close to the value of 0.47 found by author R. T. Stefani [17]. To investigate the effect of team strength on win margin, we can fit a simple linear regression of win-margin against the last-season points of each team playing. Each covariate was normalised by subtracting the mean value (57.4). We fit the following model.

$$\begin{aligned}\text{win margin} = &+0.415 \\ &+0.0315 \times \text{adj. home points} \\ &-0.0170 \times \text{adj. away points}\end{aligned}$$

The first two coefficients are found significant with $p < 0.001$, the third with $p = 0.016$. The positive intercept term corresponds to the advantage of the home team - due to positive win margin indicating a home win. The higher coefficient of the home team's points is in agreement with the stacked effect from earlier. In the context of matches between teams of similar strength, the advantage of the home side would increase linearly as the strengths of the teams increased. For instance, a match between Everton (H) and Leeds (A) (39 and 38 points resp. - at the bottom) would be predicted a home win margin of 0.166. A similar theoretical match of Manchester City (H) vs Liverpool (A) (93, 92 points - at the top) would be predicted 0.950 more goals in favour of the home team.

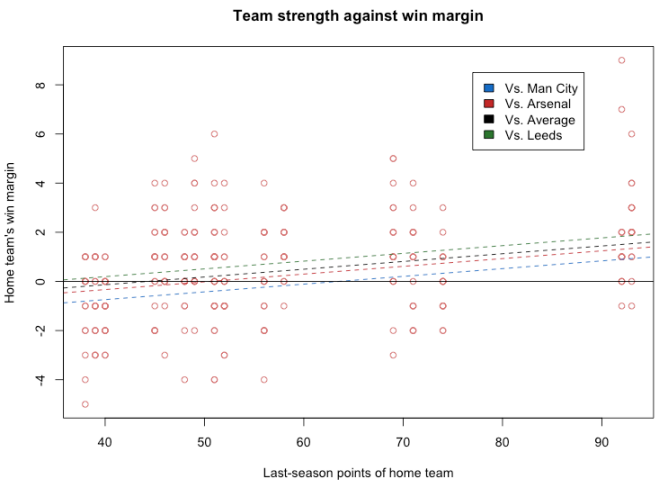


Figure 5: Plot of win margin against last-season points, with regression lines

Despite strong significance of each coefficient, this model achieves an adj. R^2 value of 0.09, suggesting that it only captures a small portion of the variation in win margin. We can visualise this fact using Figure 5. The regression lines overlaid on this scatter plot were calculated by substituting opponent points into the regression model. The teams were chosen as having the highest and lowest point values, as well as Arsenal in the middle. The large vertical spread of the data outside of the widest bounds of our regression is quite clear, hinting towards the large variation inherent in football results.

3.2.6 Other Results

No significant changes in win proportion were observed across changes in time. The time of each match was split into 4 bins and a χ^2 test was performed, finding no link with $\chi^2 = 8.4$ on 6 degrees of freedom, $p = 21\%$. This variable will be considered in later models, as there could still be interesting interactions with other variables.

Red cards are found to have a significant effect on match outcome, when awarded to the home team. Due to the low number of matches with a team awarded multiple cards, we consider only the presence or absence of red cards. A Spearman test for correlation between presence/absence of red cards against match result (as numeric value 1/2/3) shows a significant ($p = 2.9\%$) correlation for home red cards but insignificant ($p = 78\%$) for away red cards, possibly due to the lower number of those compared to home cards.

4 Logistic Regression

4.1 Modelling match outcome

In this section we hope to explore the use of logistic regression in modelling match outcomes in football. Bernoulli-distributed processes may have the 'success' chance p be dependent on different variables X. Here we have $p_i \in (0,1)$ the probability of success for the ith subject, with $\frac{p_i}{1-p_i} \in (0,\infty)$ the odds of success. Taking the logarithm of this quantity gives us the log-odds or 'logit' ($\in (-\infty,\infty)$). The logit link function therefore allows us to use regression to model probabilities, as we can write the regression model as follows:

$$\log \frac{p_i}{1-p_i} = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ...$$

Given a list of observed outcomes $\in \{0,1\}$ and corresponding values for each variable, the coefficients β can be calculated using maximum-likelihood estimation. These coefficients have a nice interpretation in terms of the odds of success, as a unit increase in the ith variable increases the odds of success by a factor of e^{β_i} An alternate way to express the model is to solve for p_i.

$$p_i = \frac{1}{1+e^{-\beta_0-\beta_1 X_{i1}+...}}$$

As mentioned in the literature review, match outcome in football is three valued, but by grouping home losses with draws we can treat it as a binary response (hence in the R code this is called `result_bin`). Using logistic regression we can then predict the chance of a home win for given values of modelled variables, and examine the effect each variable has on the odds of winning. First of all we will fit a full model, using most of the variables examined in the previous section. We will test whether red cards given to either side have an effect on the home team win chances. As the number of matches with any red cards is already fairly small we will use 1/0 valued variables denoting whether or not any were given to each side, rather than the number. Distance travelled by the away team will, as before, be grouped into a categorical variable based on whether it was above or below the median distance of 169km, or the distance between the stadia of Arsenal and Nottingham Forest. Distance could be used as a raw kilometre value, or undergo a log-transformation instead. Since it seems unlikely that a $1 \mapsto 2$ increase in distance would have the same effect on win-odds as a $100 \to, \mid 101$ increase, we won't use the raw value, though the log-transformation could have more interesting results. The points each team scored in the previous (2021 - 2022) season will be included, as will a categorical variable denoting whether or not the match was a derby. A categorical variable denoting whether the match occured in the first or second half of the season will also be included. We considered normalising the points variables. This would lead to more interpretable results when all covariates are set to zero but given the model would be equivalent as far as predictions, we decided against it.

Variable	Estimate	Std. Error	p
(Intercept)	−1.527	0.698	0.029
Away team red card (yes:1)	0.276	0.769	0.720
Home team red card (yes:1)	−1.405	0.683	0.040
Distance grouping (far:1)	0.404	0.285	0.160
Away points	−0.014	0.008	0.081
Home Points	0.033	0.008	0.000
Derby (yes:1)	−0.087	0.429	0.840
Season half (later:1)	0.370	0.259	0.150

As can be seen from the p-value column, matches being derbies and red cards awarded to away teams seem to have little effect on match outcome. For derby matches this could be due to a correlation with the `distance_grouping` variable - rival teams are generally very geographically close to one another - raising the p-value. While red cards seem like an obvious boon to the other team, the sample size here might make the true effect size hard to uncover, as only 9 matches in this data set had any red cards given to the away team (18 to home teams). Other variables have questionable significance, but there's an argument towards halving some of these p-values and treating them as one-tailed tests. The season half (`late_season`) seems like it could effect results either way, so a two-tailed test is appropriate for that variable. Red cards could be argued to be inherently good for the opposition so we will treat those as one-tailed, and the same for the 2021-2021 season league points variables and the closer/farther distance variable on the grounds that the away team would be less/more tired from the journey.

To build a more justified model, we will remove derbies and away-team red cards as variables. Though the distance grouping and season half have higher p-values, these will be kept. Season half might have some interesting interaction with other variables, and it may be worth checking whether distance becomes more relevant in the absence of the related derby variable. A call to `glm` using the formula

`result_bin ~ red_card_home + distance_grouping + opponent_points2021 + points2021 + late_season` gives a model with the following coefficients.

Variable	Estimate	Std. Error	p
(Intercept)	−1.511	0.696	0.030
Home team red card (yes:1)	−1.386	0.682	0.042
Distance grouping (far:1)	0.437	0.258	0.091
Away points	−0.014	0.008	0.073
Home points	0.033	0.008	0.000
Season half (later:1)	0.372	0.258	0.150

The AIC value of this model is 356.66, down from 360.5. Of these coefficients, home team red cards and home team points are unquestionably significant. The distance variable has a 1-sided p-value of 0.0453, and the away team points of 0.0365, justifying their inclusion. Season half still does not pass the 5% bar, and we fit a model containing interaction terms between each variable here and the season-half but none of the interactions are significant. In light of the irrelevant interactions we will try removing the season-half variable, which hasn't had any improvement in significance.

Variable	Estimate	Std. Error	p
(Intercept)	−1.322	0.680	0.052
Home team red card (yes:1)	−1.322	0.684	0.053
Away points	−0.014	0.008	0.069
Home points	0.033	0.008	0.000
Season half (later:1)	0.423	0.257	0.100

The AIC of this model is 356.75, ever-so-slightly higher than the previous. We can calculate a McFadden R^2 for this model of 0.078, down slightly from the values of 0.085, 0.084 for each of the previous two models. This R^2 is quite low, suggesting much of the variation in the data is yet unnaccounted for. A McFadden R^2 in the 0.2 to 0.4 range is considered an excellent fit [18]. The significance of the distance grouping is right on the 5% one-sided borderline, but for the sake of a more interesting model we decide to keep it. Given all the other variables are significant we will proceed with this model. The coefficient of home team cards is remarkably similar to the intercept term, −1.3221 vs −1.3218, but as far as we can tell this is just coincidence.

Taking exp on each coefficient we can see the effects of each variable on the odds. Red cards are associated with a reduction in the odds of a home win by a factor of $\exp(-1.322) = 0.267$, which makes sense given the home team would be facing at a one-player disadvantage for some portion of the match. As a heuristic, we can check this value as follows. Home teams won 129 of 272 total matches, and of the 15 where they were given red cards they won 3. The odds of each event (win in general, win with red card) are thus 0.902 and 0.25, and we see that $\exp(-1.322) \times 0.902 = 0.24 \approx 0.25$.

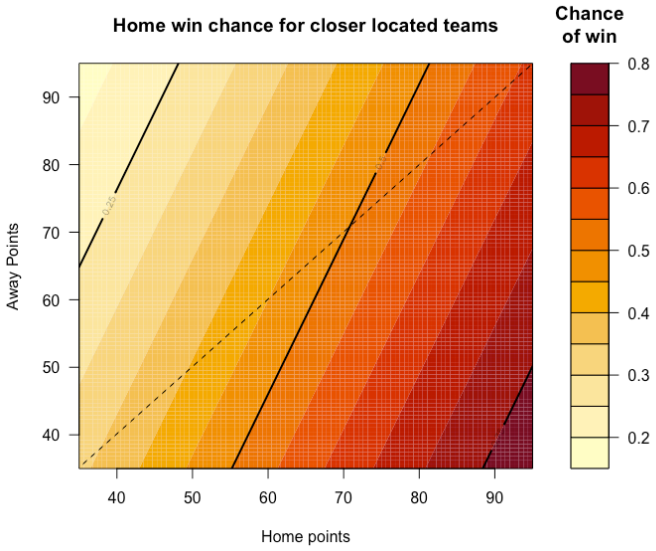
Each additional point the home team achieved in the previous season corresponds to a 1.03 times greater odds of winning, whereas each such point by the opponent multiplies the odds by 0.986. We can see confidence interval bounds on these odds ratios in the table below. The mean column gives point estimates for the odds multiplier $\exp(\beta_i)$ of each variable, and the lower and upper columns give 95% confidence interval bounds on these.

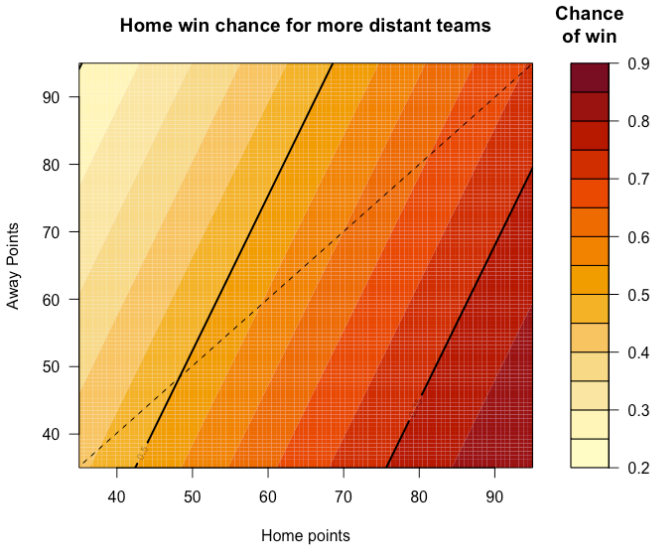
Variable	Lower	Mean	Upper
Red Card Home	0.070	0.267	1.011
Away Points	0.970	0.986	1.001
Home Points	1.017	1.034	1.050
Distance	0.922	1.526	2.526

The small number of matches with red cards given out leads to the bounds on this multiplier being quite wide. The 5.2% two-sided p value given earlier is seen here as the uppermost bound ($z = 1.96$)on this quantity only just exceeds 1. The bounds in general are quite wide. In the case of red cards this is likely a consequence of the low sample size. For the other variables this could be because of a slight lack in predictive power. Points as a proxy for strength isn't a perfect system, as a team's performance will naturally change between and through seasons, and also because the points aren't solely determined by strength, for instance they can be deducted when teams break rules.

As we have 4 variables it's difficult to plot the probability of a home win, however assuming no red cards are given we can provide two contour plots of Pr(Home Win) against the points of each team. In this first plot we set the distance grouping variable to 0, representing a closer-located away team. In Figure 6 we set the distance grouping variable to 0, representing a closer-located away team, and in Figure 7 we set the variable to 1. The bounds on each axis correspond to just either side of the minimum and maximum points values attained in the previous league season, 38 and 93. Three solid lines have also been plotted, each a level curve at which a match has a home win proba-

bility at 25, 50, 75%. The dotted line is $x = y$, where the 'strength' of each team is equal.





The larger coefficient of home team points vs away team points is visible here as the steeper slope of the level curves compared to $x = y$. This corresponds to stronger home teams expecting a higher chance of victory even when playing against equal opposition, as compared to weaker home teams. We can see the effect that location has as well, with 25% home win chance line being almost invisible when the home team plays more distant opposition.

Home advantage is tricky to quantify using this model due to the choice of response variable, which groups the neutral outcome of a tie with the negative outcome of loss. To make this somewhat easier to reason about we can try to fit a different class of model, one that can be fit against all three possible outcomes.

4.2 Ordered logistic regression models

Ordered logistic regression models (OLRMs) are a way to generalise logistic regression, and they can be used to model situations with > 2 ordered outcomes. OLRMs have quite a similar setup to regular logistic regression. Binary logistic regression can be interpreted as a so-called latent variable model, whereby we define an unobserved variable Y^* corresponding to the observed outcome Y as follows. Let $Y^* = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \varepsilon$, with ε a standard logistic error term and X_i the ith variable. Coefficients β are estimated such that Y^* is positive when the observed outcome $Y = 1$, and negative otherwise. Y^* can be seen as a continuous version of the discrete outcome, on the log-odds scale. We can suggestively rewrite the latent variable formulation of binary logistic regression as

$$Y = 1 \Rightarrow Y^* - \beta_0 = \sum_i \beta_i X_i > -\beta_0$$

The quantity $Y^* - \beta_0$ therefore corresponds to different observed outcomes based on how it compares to $-\beta_0$, which acts as a threshold value.
In ordered logistic regression, a similar process is followed. Instead of observing outcomes $Y \in \{0, 1\}$, we observe a number of ordered outcomes, labelled $\{1, ..., k, ..., N\}$, and find coefficients β_i, μ_k such that $\mu_{k-1} \leq Y^* = \sum_i \beta_i X_i \leq \mu_k$ corresponds to the observed outcome $Y = k$. The values μ_k therefore take the place of β_0 as threshold values marking different outcomes. The odds and probabilities of each outcome can be calculated using the following formula, reminiscent of $Y^* - \beta_0$ in the previous paragraph:

$$\text{logit}(\Pr(Y \leq k)) = \mu_k - Y^*$$

Coefficients:			
Variable	Value	Std. Error	t value
Home red card	−0.915	0.484	−1.892
Home Points	0.027	0.008	3.548
Away Points	−0.006	0.007	−0.813
Distance group	0.266	0.232	1.147
Intercepts:			
Boundary	Value	Std. Error	t value
Loss|Draw	0.068	0.627	0.108
Draw|Win	1.405	0.632	2.223

Here we will try once more to fit a model to match outcome in football, this time respecting all three possible outcomes. We can fit an OLRM in R using the `polr` function in the `MASS` package. For the sake

of comparison we'll use a model fit against the same variables as the previous. Coefficients can be found in the table above. The low t-values given clearly hint towards a low-confidence in these estimated values, a few of which are lower in magnitude than their standard errors. Nonetheless we can see the outcome this model would predict for a hypothetical game. For a game between Chelsea (74 pts, home) and Arsenal (69 pts) with no red cards we would see

$$\begin{aligned}\widehat{Y^*} = \sum \beta_i X_i = (&0.027 * 74 \\ &-0.006 * 69 \\ &+0.266 * 0 \\ &-0.915 * 0) = 1.677\end{aligned}$$

This value 1.677 is greater than 1.405, hence the match is most likely a home win. We can also look at the probabilities predicted for this game to fall into each outcome. Given the boundaries μ_k demarcating each outcome are $-\infty, 0.068, 1.405, \infty$, we can extract each probability as:

$$\begin{aligned}\Pr(Y = k) &= \Pr(Y \leq k) - \Pr(Y \leq k-1) \\ &= \text{invlogit}\left(\mu_k - \widehat{Y^*}\right) \\ &- \text{invlogit}\left(\mu_{k-1} - \widehat{Y^*}\right)\end{aligned}$$

Where invlogit is the inverse logit function, $x \mapsto (1 + \exp(-x))^{-1}$. Thus the probability of a home loss would be $\text{invlogit}(0.068 - 1.677) - \text{invlogit}(-\infty - 1.677) = 0.17$ (taking limits where needed), similarly $p = 0.27$ for a draw and 0.56 for a home win.

In order for this class of models to be appropriate, the data must satisfy the proportional odds assumption. This assumption mandates that the odds ratio between outcomes "k_1 or less" and "k_2 or less" must remain constant over different values of the independent variables. There are a few suggested practical methods for assessing whether this assumption holds over a data set. An online search suggests either graphical methods via plotting cumulative probability against each predictor [19], performing a likelihood ratio test between the OLRM and a multinomial regression fit using the same variables, or the Brant test [20], implemented in R in the `brant` package.
Multinomial logistic regression models are similar to OLRMs. In essence, seperate logistic regression models are fitted for each outcome, the results of each model giving the odds of a given subject falling into each outcomes. This allows for modelling outcomes that aren't ordered, at the cost of estimating multiple coefficients for each variable. The likelihood ratio test then looks at whether this model fits the data sufficiently better. The Brant test takes this same approach, but performs seperate tests on each variable.

We can perform the LRT fairly easily. The test has null hypothesis that the odds are indeed proportional. We find deviance values of 553.04 and 537.47, with degrees of freedom 12 and 20 for the OLRM and multinomial models respectively. This corresponds to a χ^2 test statistic of 15.55 on 8 d.f., higher than the critical value of $\chi^2_{0.05} = 15.51$, hence we should assume that our data do not satisfy the proportional odds assumption. The Brant test confirms this result.

Test for	X2	df	p
Omnibus	17.040	4	0.000
Home red cards (yes:1)	0.600	1	0.440
Home points	7.240	1	0.010
Away points	5.620	1	0.020
Distance grouping (far:1)	2.680	1	0.100

The null hypothesis is again that the odds are proportional, and we can see the offending variables are the points of either team. We can try fitting a new model, without the raw points values and instead using the above/below median grouping by points variable that we saw in the exploratory analysis section.

Test for	X2	df	p
Omnibus	6.030	4	0.200
Home red card (yes:1)	0.730	1	0.39
Home point grouping (lower:1)	1.010	1	0.31
Away point grouping (lower:1)	1.69	1	0.19
Distance grouping (far:1)	2.29	1	0.13

And we see that the factor model can satisfy the assumptions required for OLR. This model ignores a lot of information as a result of disregarding the more granular data in the raw points values, however it meeting the assumptions does raise our confidence in the results of the model.

4.3 Forecast Performance

We can test the forecast performance of all the models seen in this section. To avoid testing on the same data the models were trained on, we split our current dataset in two, 20% into the testing portion, assigned at random. New models with the same specifications were trained on the remaining 80% of the data. The coefficients in these testing models are fairly similar

to the previous so we won't go into detail on them, though for obvious reasons the errors are wider.

This means the relevance of forecasting results here is murky. If we wished to simulate actual predictions, say predicting future matches in the league, we could have split the matches at a particular date and time. Though the significance of season time was shown earlier to be low, we thought it might bias the results to some degree. We could also have used the previous models to predict matches in the following league season. Between the change in teams due to promotion/relegation, as well as the even-more out of date points values, we decide to stick to the 2021-2022 season.

Out of 62 matches in the testing set, we see 26 wins, 20 ties and 16 home losses. In this sample tied results are very overrepresented, in general being rarer than losses, but the win:loss ratio remains similar, 1.63 vs 1.65 in the season overall.
For the binary logistic regression model we see the following results.

Pred/Actual	Home Win	Not
Home Win	15	11
Not	11	25

So 40 out of 62 matches are correctly predicted. The model seems to err very symmetrically, though the tie-bias in the data is likely counteracting the bias inherent in the model. Below is the table for the ordinal logistic regression model that failed the proportional odds test earlier, included for comparison.

Pred/Actual	Win	Tie	Loss
Win	25	18	13
Tie	0	2	1
Loss	1	0	2

And for the categorical (factor) model that passes the Brant test, we see the following predictions.

Pred/Actual	Win	Tie	Loss
Win	25	19	15
Tie	0	0	0
Loss	1	1	1

We can see that both OLRMs are heavily skewed towards predicting home wins. Given the similarity of results we probably can't attribute this to the difference in variable choices, continuous or categorical descriptions of points. The large standard errors we saw earlier on the μ_k threshold values are probably to blame. While it would lack any theoretical justification we could probably see more accurate results by increasing the values of both thresholds (between loss and tie, tie and win) as well as the distance between them.

A better solution might involve an entirely different quantification of "strength". Restricting ourselves to data also from the previous season, perhaps the team's ranking may be better, or strength could be calculated using the α values estimated in a Bradley-Terry like model trained on last-season performance. Of course, incorporating bookmaker's odds or even just results from previous matches in the season would likely yield much more capable models, so if we wanted more positive results those are the directions in which we'd be most likely to look.

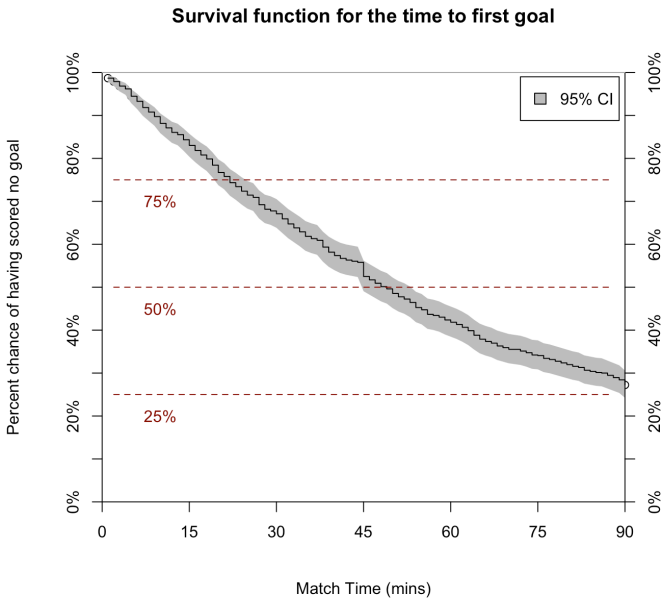
5 Survival Analysis

5.1 Interactions between variables and survival time

The time to first goal (TTFG) is a statistic of interest for both bookmakers and bettors. Many bookmakers offer the ability to bet on a timeframe in which the first goal will be scored, usually dividing the match into 10 or 15 minute intervals. Similar bets can be placed on whether a goal will be scored before/after certain points in the game, particularly around the number of goals scored in the first/second half.

In this section we hope to answer some questions about the TTFG in football matches. We will examine its dependence on various factors, such as comparing home teams vs away. The Kaplan Meier plot is a We won't investigate any continuous variables until later as these aren't amenable to Kaplan Meier fitting, hence all the over/under factor variables. Also of note is that the data set in this section is structured differently to the previous. In the logistic regression section we looked at data where each observation corresponded to a match, in this section each observation corresponds to the match from the perspective of each side, home and away. This makes model fitting easier since we have one goal time in each observation but we ought to be careful examining variables with side-dependent effects, as any effect that benefits home sides as much as it hurts away sides will have no effect overall.

Figure 8 shows a Kaplan-Meier survival curve for first-goal events across every match in the 2022/23 season, including data from both home and away teams, Notable in this plot and most of the following ones is the sharp drop at 45 minutes. This happens because goals occuring in injury time at the end of the first half are counted as occuring in the 45th minute, thus raising the number of goals falling into this bin. All highlighted regions represent 95% confidence intervals, based on the estimated standard error of survival chance.



A quarter of games have their first goals within 22 minutes. The median time for first goal is 49 minutes, slightly into the second half. The third quartile value is not reached, as 72.8% of teams end the game without scoring, just shy of three quarters.

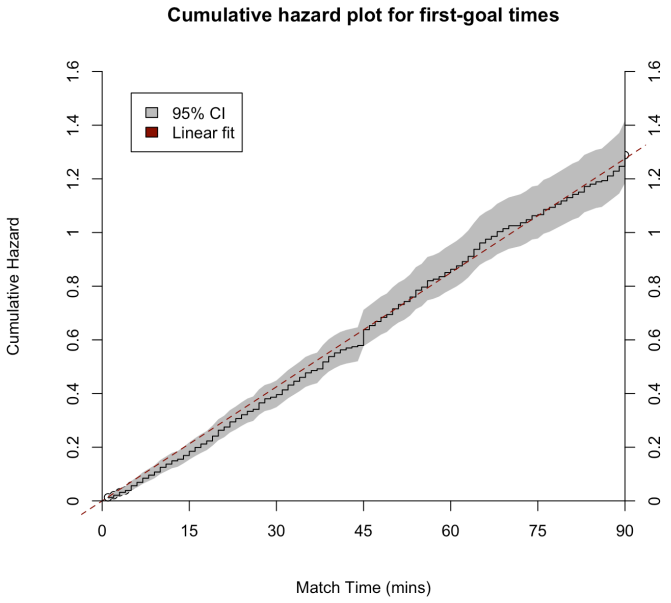


Figure 9 shows a cumulative hazard plot (CH plot) for times to first goal, along with a dashed line showing a straight-line fit. Apart from the jump at 45 minutes, it is clear that the graph is very close to linear, a visual suggestion that the hazard is close to constant throughout the match.

5.1.1 Interaction with game side

As shown in Figure 10, TTFG is clearly different for home sides vs away. The median first goal times are 45 and 53 minutes for home and away sides respectively, with lower quartiles in the 21th and 23rd minutes. The upper quartile TTFG for home sides lies in the 84th minute, but among away sides only 67.1% manage to score by the end of the match.

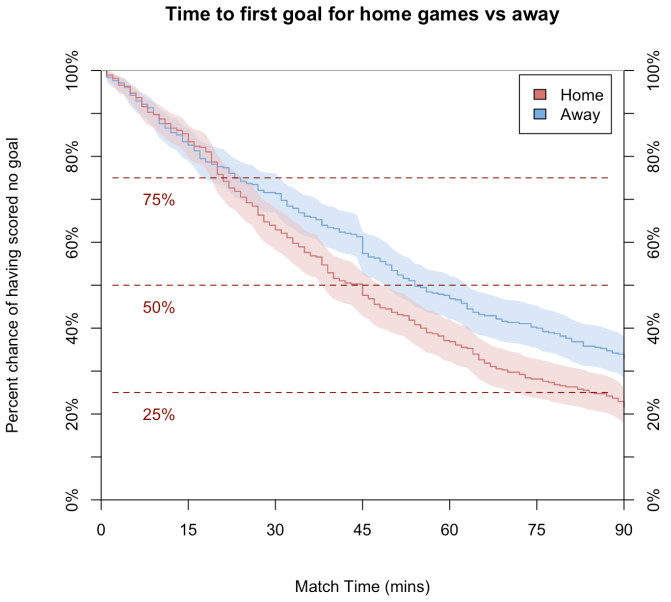


Figure 10: Survival curves for each side

To ascertain whether these subgroups are really different from one another, we can first examine their respective cumulative hazard plots.

Figure 11 shows CH plots for both home and away sides, as well as the difference in the logarithms of each observed CH function. The red and blue dashed lines show best fit lines for each. The black curve gives the log of the hazard ratio (HR) between the groups, or equivalently the log of the difference between the two cumulative hazards. A constant log difference line implies the CH functions are proportional to one another. An assumption underlying many models in survival analysis is that of proportional hazards, where the hazard (equivalent cumulative hazard) functions differ by a constant multiplicative factor. Among other things that we'll see later, this assumption justifies the use of more powerful log-rank test versus the

Wilcoxon-Gehan test for difference in hazard between the groups. In order to calculate the log HR the CH for one treatment group has to be interpolated to match the number of observations seen in the other, but this should not change its shape.

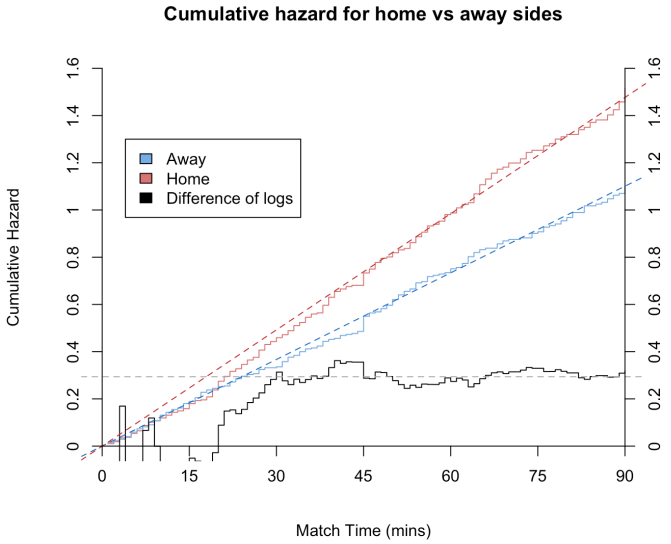


Figure 11: Cumulative hazard plots for home and away sides, and their log difference.

The log HR appears fairly constant after around 30 minutes, but before this it varies quite a bit, even changing sign at one point. This could be due to away teams playing more aggressively in the opening of a match, to gain an early lead; the aforementioned authors Nevo and Ritov observed a similar effect in their paper. As we will see a similar effect is present for many variables, with different groups indistinguishable in the opening minutes of the match. Some portion of this effect is possibly numerical, any effect on play due to a change in some variable will have had less chance to accumulate near the start, hence the groups will be closer together and their ratio will be more effected by noise.

For this reason we will assume for this and similar examples that the hazards are proportional, and use the log-rank test, implemented in R using `survdiff(..., rho = 0)`. The test shows that these populations differ significantly, with $\chi^2_{\text{df}=1} = 11.6, p = 0.0005$.

5.1.2 Interaction with team strength

We can plot seperate KM curves for each team grouping, by whether they attained above or below median points in the previous season. As we saw in the logistic regression section, these variables in their raw numerical form were problematic for the proportional odds assumption of ordered logistic regression.

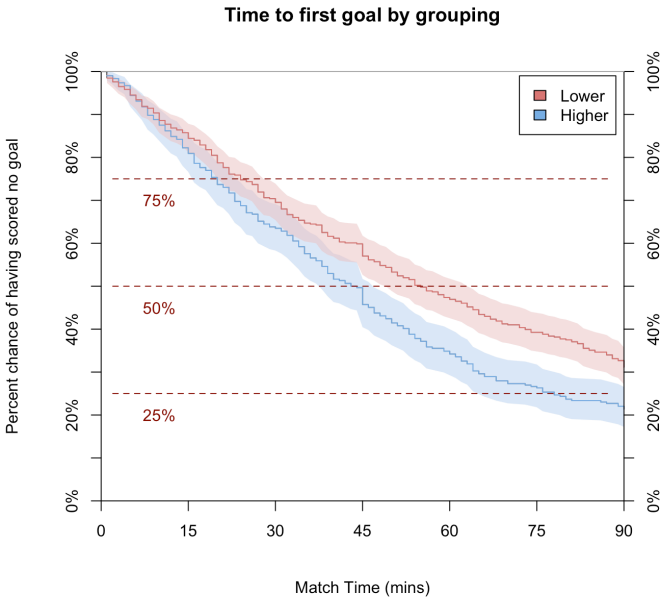


Figure 12: Survival curves by team grouping

The curves in Figure 12 are superficially similar to those in Figure 10. The quartiles for higher grouped teams are in the 19th, 43th and 76th minutes, for lower teams the 23rd, 54th and 90<th minutes, with 31.1% of these lower grouped teams failing to score.

Much like with home vs away sides, the CH curves are fit quite well by the lines, and the log difference is fairly constant after 30 or so minutes. The arm for lower grouped teams appears somewhat concave through the second half of the match, though this a fairly minor deviation.

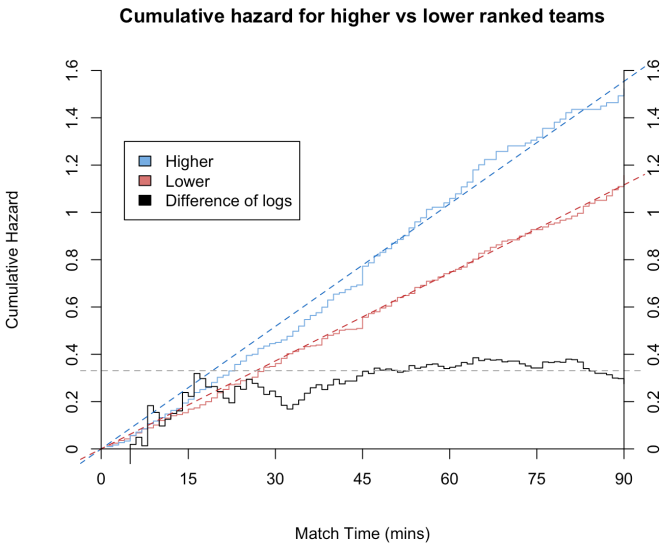


Figure 13: Cumulative hazard by team grouping

For the same reasoning as before, we perform a log rank test, showing this split has real effects on TTFG, giving $\chi^2_{\text{df}=1} = 11.9, p = 0.0006$.

5.1.3 Interaction with other team's strength

Intuitively the strength of the other team may also affect the rate at which a team can score. As in the previous section we'll look at the binary grouping by last-season points here to determine stronger and weaker teams.

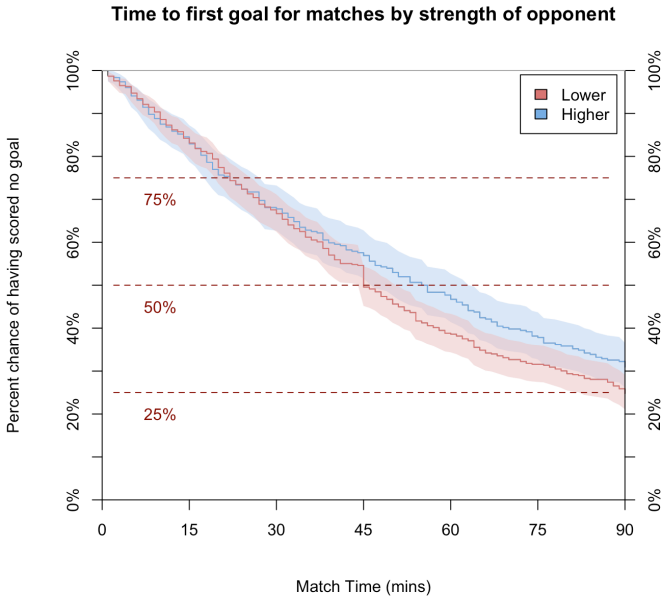


Figure 14: Survival curves by opponent's grouping

The survival curves in this example are much closer together than the previous, but the effect still seems visible. As with many of the other variables looked at here, the curves are much more clearly distinct towards the second half of the match. Unlike the others, this distinction appears later, at around the 45 minute mark. The quartile values of TTFG are at 21, 45 and 89 minutes for teams playing a weaker opposition, and 21, 55, 90+ minutes otherwise, with 30.9% of teams facing a tougher opponent failing to score by full time.

The log difference line again appears constant nearer the end, as the hazard ratio appears to rise towards and settle at $\approx e^{0.2}$. The relatively larger shifts in the observed hazard ratio mean we will test the difference here using the Wilcoxon-Gehan test, resulting in a chi-square value of 1.9 on 1 d.f., so $p = 0.2$. This is above the usual threshold but given the interaction between the strengths of either side, seen in previous sections, it may be worth investigation in a fuller model.

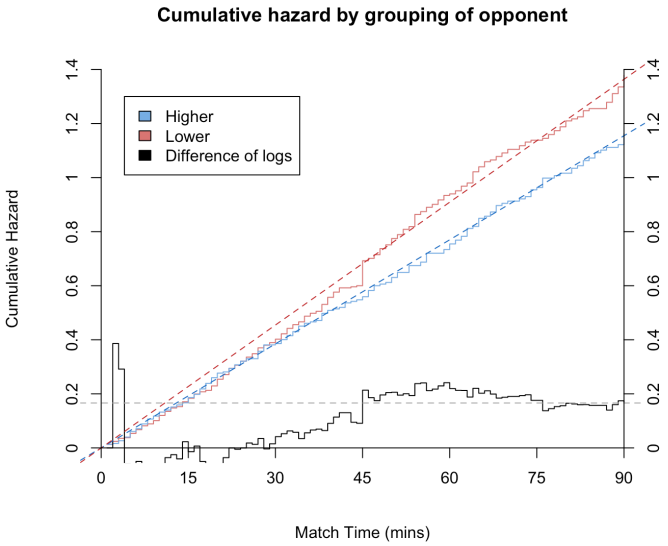


Figure 15: Cumulative hazard by opponent's grouping

5.1.4 Interaction with distance travelled

Restricting our attention to away sides, we fit survival curves to TTFG based on whether each away side travelled more or less than the median distance to attend a match. As mentioned at the beginning of this section we do focus on away sides as the effect of this variable presumably depends on the side. In the next section we will fit this as an interaction term.

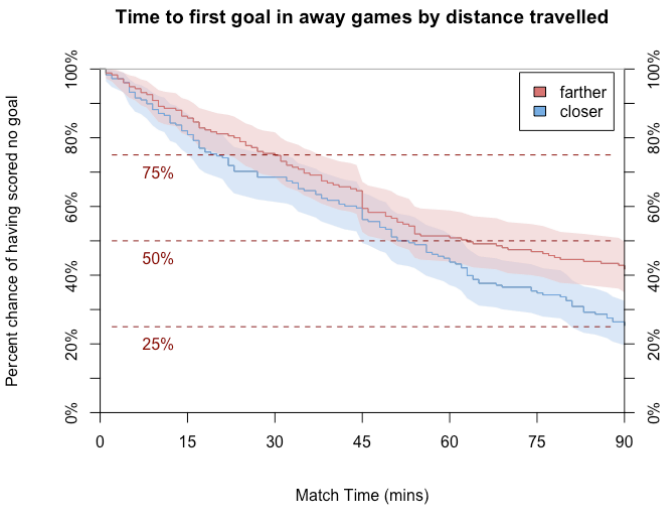


Figure 16: Survival curves for away sides by distance travelled

Figure 16 shows the results, again quite clearly distinguishable. Likely due to the inherent disadvantage suffered by every team examined here, neither curve quite reaches below the 25% line by 90 minutes, with teams who travelled a longer distance failing to score 41.1% of the time and closer teams 24.7% of the time. The lower quartile for first goal time lands in the 19th

and 27th minute for near and far teams respectively, with medians at 50 and 62 minutes.

Figure 17 shows that the CH plots for this variable don't fit the lines quite as cleanly. Some measure of this is likely due to the smaller sample - we've excluded results for home sides - and this is shown too in the confidence intervals in Figure 16.

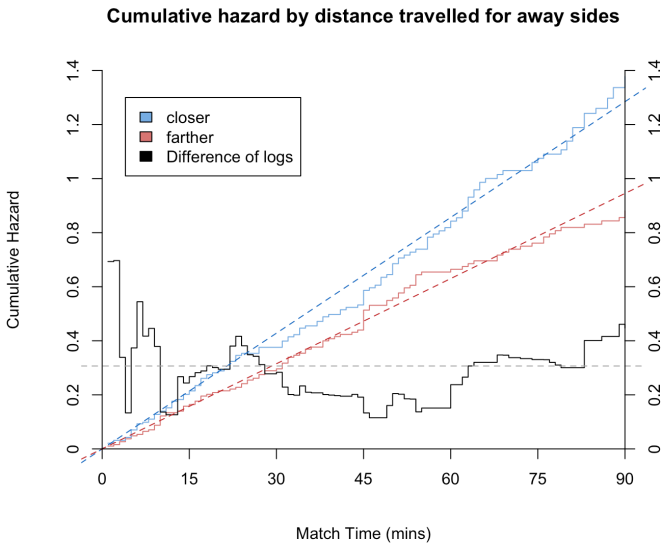


Figure 17: Cumulative hazard by distance travelled

Scoring rate appears to pick up at the end of the game for closer-located teams, and unlike with our previous two variables we see that the distinction between the groups near the start of the match is much wider, with the log difference line being positive (if very bumpy) for almost the entire 90 minute window. The W-G test shows $\chi^2_{\text{df}=1} = 5.8, p = 0.02$, leading us to take this as a significant difference.

5.1.5 Interaction with derby matches

A 'derby' game is common terminology for a football match between two rival teams. Usually both teams are local to one another, maybe representing neighbouring towns or nearby city districts. These games are often considered to be higher stakes for fans of each team. In some particular cases this has boiled over into incidents of violence or property destruction. Due to the fact that derby games are fought between geographically close teams, the effect found earlier where away teams score worse when travelling farther might lead these games to being more closely matched than games in general.

Our definition of what games constitute derbies can be found in the data collection section earlier. Perhaps due to the much smaller size of the derby group (about 60 games in 380) the estimates given by R are much wider, visible in the confidence intervals in red in Figure 18. The survival curves here seem very similar, with the point estimates not too distinct and one line contained completely within the CI of the other.

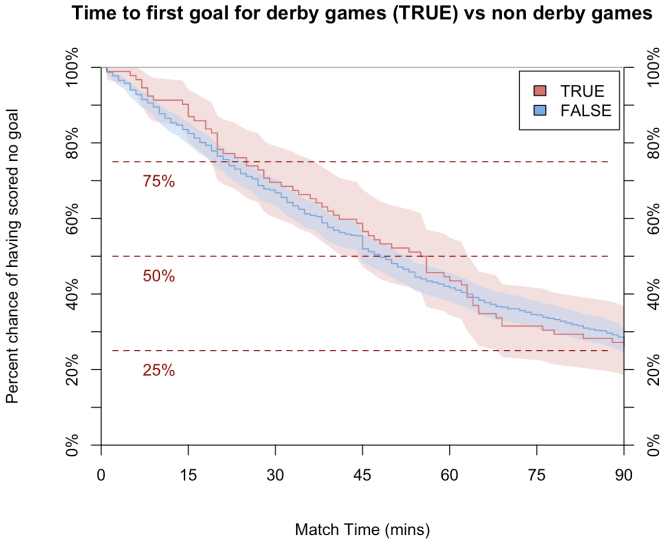


Figure 18: Survival curves for (non-/)derby games

This similarity is just as visible in the CH plots, where both lines follow very similar paths. This could be a symptom of the smaller sample size, as that is likely what lends the curve for derby games to be so jagged.

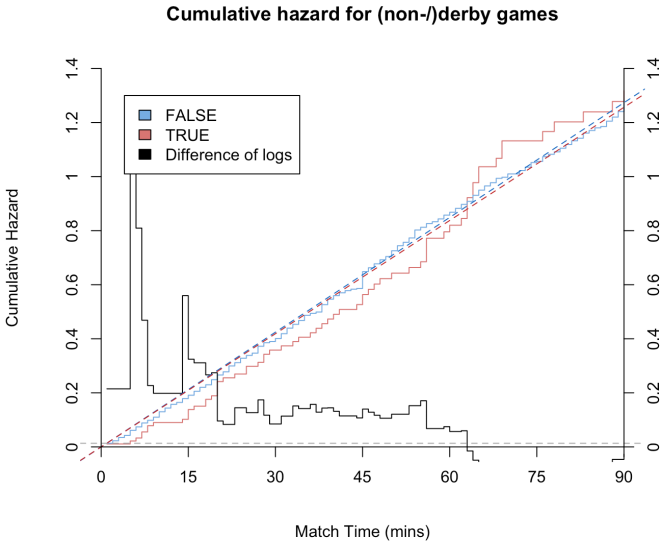


Figure 19: Cumulative hazard in (non-/)derby games

The estimated log difference is very close to zero, and the observed log difference doesn't follow it closely at any point on the plot. Due to the erratic log-difference curve, we will confirm our visual analysis with the Wilcoxon test, which gives $\chi^2_{\text{df}=1} = 0.5, p = 0.5$.

5.1.6 Other interactions

An interaction with the day of the game was tested. No significant link was found between date/time of kickoff (for this we use the Unix timestamp, an integer value which gives the number of seconds since 1/1/70) and TTFG, nor was a link found for a similar factor variable classifying whether the match occured earlier/later in the season.

Red cards, much like derby games, are scarcely represented in the data, hence much like derby games no significant effects were found in tests, even when restricting the data to only one side. In some literature (see [9]) the authors use the match time at which red cards are given to investigate the change in hazard before and after the event but we don't have this data. The overall effect of red cards on first-goal time may be murky, since naturally it means a team is playing with one fewer player for some time, however there may be a confounding effect whereby a more aggressive team would score quicker goals and commit more rule-breaking behaviour. We will investigate this as an interaction term.

5.2 Fitting parametric models

The R function `flexsurvreg` can fit parametric survival models of two kinds, accelerated failure time (AFT) and proportional hazards (PH). AFT models focus on the effects of each variable on survival times, so the survival time of the kth subject is distributed as $T_k = e^{\eta_k} T_0$, with $\eta = \sum_i \beta_i X_i$ the linear predictor and T_0 follows some appropriate baseline distribution. A unit change in the linear predictor thus "speeds up time" for the kth subject by a factor of e.

By contrast PH models focus on how each variable affects the hazard, so $h_k(t) = \exp(\eta_k) h_0(t)$ gives the hazard of the kth subject as a multiple of a baseline hazard. A unit increase in the linear predictor thus "increases the risk" of the event of interest by a factor of e. For parametric models this h_0 has a closed form, for instance an exponentially distributed survival time correponds to a constant baseline hazard. Weibull models (and therefore exponential models) can be used in either of these frameworks, as Weibull distributions satisfy both properties. For the sake of comparison with the popular Cox regression model, we will take the proportional hazards approach.

Having examined the effects of a few variables on TTFG alone, we will try fitting two parametric models to our data, one exponential and one Weibull. In the logistic regression section we saw that the raw points values didn't follow the proportional odds assumption. As both the exponential and Weibull models follow a proportional hazards relationship, we may see a related effect here. The first step will be to remove insignificant variables, and then to check that the model fits are appropriate. Away team travel distance as a raw kilometer value was tested but found insignificant, so much like with the ordered logistic regression model we will use the closer/farther variable. The model using raw distance values reduced down (after removal of insignificant variables) to a model strictly nested inside the one we'll soon arrive at, hence we won't include it here.

```
s ~ home_or_away*(red_card_home + red_card_away
+    distance_grouping)    +    points2021    +
opponent_points2021
```

Given the large amount of data in the model selection process, relevant R output can be found in the appendices. We first remove any insignificant interaction terms to arrive at our 2nd group of models, then to arrive at our third and final set of models we remove any variables still insignificant.

The results of this process are in the following table. Both final models use the same set of variables.

Variable	Estimate	S.E.
Shape γ	1.145	0.050
Scale α	0.007	0.002
Home (1) vs Away	0.037	0.140
Closer vs Farther (1)	−0.460	0.148
Points last season	0.010	0.003
Opponent's Points	−0.007	0.003
Side*Distance	0.571	0.202

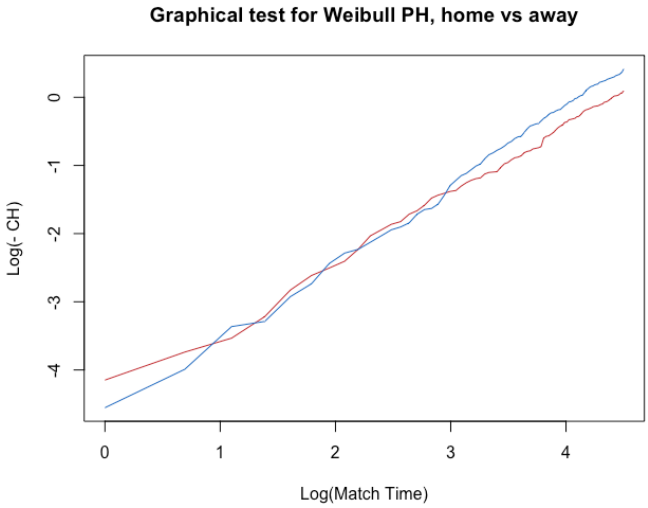
And for the exponential model:

Variable	Estimate	S.E
Rate λ	0.013	0.003
Home (1) vs Away	0.030	0.140
Closer vs Farther (1)	−0.442	0.148
Points last season	0.010	0.003
Opponent's Points	−0.007	0.003
Side*Distance	0.548	0.202

All coefficients are significant to a 95% level, except the home/away factor, which is included as its interaction with distance is significant. Unlike in the logistic regression section, red cards seem to have less of an effect here. This could be because first goals skew

towards the early game, meaning they will often have already occurred by the time a red card is given, unlike match outcome which is affected by events throughout the full course of the match. The opposite signs on the distance variable and the distance interaction can be explained as the effect of travel distance differing for each team side. Coefficients in each model are broadly similar, as could be expected for similar baseline distributions. The shape parameter estimated for the Weibull-based model is greater than one, indicating an increasing hazard.

There are a number of methods for assessing Weibull PH model suitability. If the model depends on some categorical variable, one can plot $\log(-\log(S(t)))$ against $\log(t)$ using the Kaplan Meier survival functions $S(t)$ estimated for each strata. If the underlying distribution is indeed Weibull, then under the proportional hazards assumption this plot should show a number of parallel lines.



In Figure 20 we show an example of such a test, for the home vs away variable. Most of the model observations are contained in the $3 < \ln(t)$ range, hence the clearer distinction there. We can see the same effect here as in the non-constant log HR curves earlier, where the curves seem hard to tell apart in the earlier parts of the game. Nonetheless we can at least see that the lines appear parallel around the top right. This method is less useful for continuous variables like points.

There exist residual methods for assessing the fit of parametric survival models exist, such as Cox-Snell, martingale, and deviance residuals. We will just examine the Cox-Snell residuals.

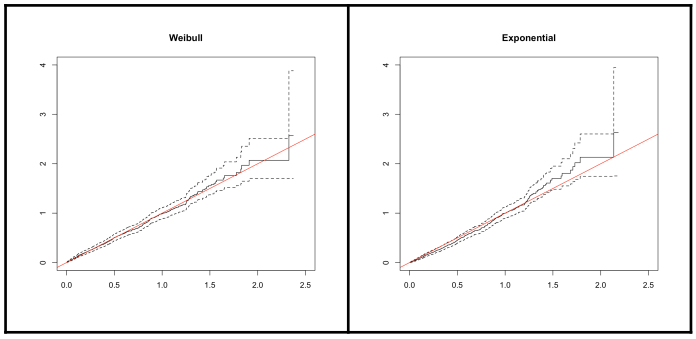


Table 1: Plots of Cox Snell residuals

In the Q-Q plots above, we plot Cox-Snell residuals against theoretical ~exponential(1) quantiles, with the $y = x$ line overlaid in red.

Non-censored Weibull-distributed data have, in theory, exponentially distributed Cox-Snell residuals. [21]. We see that both models fit the data quite well. For both models the smaller residuals seem almost perfectly exponentially distributed. At the upper end, the Weibull model seems to fit a bit closer, whereas the exponential model seems to have residuals lying above the line.

To examine which of the models fits better we can perform a likelihood ratio test. As the same variables turned out to be significant, and the exponential is a special case of the Weibull model, these models are nested. Calculating $-2(l_2 - l_1)$ we find a value of $\chi^2 = 9.07$. This is on 1 df, as Weibull models estimate exactly one extra parameter, the shape γ. This is above the 1% χ^2 critical value of 6.635, hence we can conclude that the extra flexibility of the Weibull model does improve the fit somewhat.

We can quantify the estimated hazard mulitipliers for different categories of teams over the baseline hazard. While the differing coefficients of each team's points indicate a changing relationship of the hazard as points vary, we will just assume the two teams have an average number, 57.2.

	Home	**Away**
Closer	1.25	1.20
Farther	1.39	0.76

The advantage of home teams is clearly seen by comparing the first column to the second. The interaction with distance also shows, as away team performance almost matches that of home teams when they don't have to travel as far

5.3 How does a Cox regression compare?

The linear fits to the Kaplan Meier curves shown in the univariate analyses seemed fairly close, but the discontinuous jump at 45 minutes and seeming non-linearity throughout parts of the curves may hint that a more general approach could be useful. The Weibull models we just fit can accommodate a changing hazard, but they only have limited parameters through which to describe the full shape of the curve - their cumulative hazard has the form $H(t) \propto t^{\gamma}$. For some curves, such as in Figure 11, the early sections appear linear, and these are followed by a slightly concave shape. In order to account for these features, as well as the jump in hazard at 45 minutes, we can use Cox regression.

Cox regression still assumes proportional hazards, that is, the shape of the curve must be more or less the same for e.g. home teams and away teams, but they may differ by a constant multiplicative factor. The advantage of this is that the shape can be abstracted away, using a baseline hazard $h_0(t)$ of which the hazards for each treatment group are multiples. This means we aren't tied to a power-function cumulative hazard curve like a Weibull model, however it means that the survival curve may not be smooth, and we lose the niceties of a parametric distribution such as closed-form expressions for quantities of interest.

Since we've already determined the variables that are significant for the parametric model, we will fit a Cox model against the same covariates, and we find the following coefficients.

Variable	β	p
Home (1) vs Away	0.0350	0.802
Closer vs Farther (1)	−0.460	0.002
Points last season	0.010	0.000
Opponent points	−0.0073	0.020
Side * Points	0.574	0.005

The coefficients here are very similar to those in the Weibull fit, providing some more evidence that goal times follow a Weibull distribution quite well, as the less constrained baseline hazard function here still results in similar coefficient estimates.

As mentioned, the Cox model requires the proportional hazards assumption to hold. We can check this in R using `cox.zph()`. The output of this function is as follows:

```
> cox.zph(coxreg)
                                chisq df     p
home_or_away                   0.5673  1 0.451
distance_grouping              0.0699  1 0.792
points2021                     0.6350  1 0.426
opponent_points2021            2.0605  1 0.151
home_or_away:distance_grouping 3.3777  1 0.066
GLOBAL                         9.0028  5 0.109
```

This test uses residuals from the models to test against the null hypothesis that hazards are indeed proportional. As we can see, none of the tests for individual variables, nor for the models as a whole, gives enough evidence to reject the null hypothesis, though the interaction term gets close.

If (e.g.) the home-advantage, expressed as a hazard ratio between home and away teams, were larger towards the end of a match, this would violate the proportional hazards assumption. The Schoenfeld residuals s_k at time t_k are calculated as the differences between the covariate values of the subject having an event at time t_k, minus the mean covariate values of those subjects still at risk, who haven't had the event yet [22].

These can be shown to have expected values proportional to the difference $\beta(t) - \beta$ between a time-varying coefficient and the constant Cox regression coefficient [23]. Therefore a trend in the Schoenfeld residuals implies that the model should perhaps be re-specified with a time-dependent coefficient.

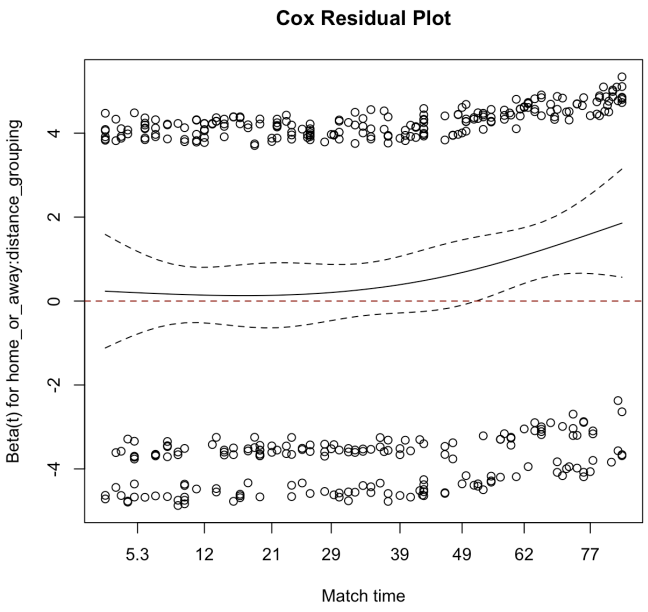


Figure 21: Schoenfeld residuals against transformed match-time

In Figure 21 we plot scaled values of s_k against time as the solid line. Match time is rescaled according to event density, so the intervals between x-axis ticks get longer.

We can see that the black line rises toward the end of the match, indicating an increased effect on hazard ratio between the groups. We could conjecture this is caused by away teams who travel farther experiencing more fatigue as the game goes on, giving their opponents an edge, though the trend isn't strong enough to be statistically significant (and we don't have the data to confidently make causal arguments like this). Similar plots can be made for the other variables, though they show no trends of a similar magnitude.

Reassured that our Cox regression fit is appropriate, we can make some comparisons to the parametric model. To pick an example we can look at the hazard ratio for Arsenal (69 points last season) playing at home against Newcastle (49 points, 394km away), over the baseline hazard. For our Cox model we see

$$\begin{aligned}\frac{h_1(t)}{h_0(t)} &= \exp(0.0350 \times 1 - 0.460 \times 1 + ...) \\ &= \exp(0.515) = 1.673\end{aligned}$$

And the corresponding ratio for Newcastle in this match is 0.639, giving a hazard ratio between the teams of 2.620. Taking each of these values to be their lower (upper) 95% confidence bounds, we find a lower (upper) estimate for this hazard ratio to be 1.350 (5.0837)

We can calculate the same ratio for our earlier parametric models. The hazard function for the ith subject can be parametrised as

$$h_i(t) = \alpha\gamma t^{\gamma-1}\exp(\beta_1 X_{i1} + ...)$$

Then the hazard ratio between teams 1 and 2 is:

$$\frac{h_2(t)}{h_1(t)} = \exp\left(\sum_j \beta_j(X_{2j} - X_{1j})\right)$$

The between-teams hazard ratio for our earlier Weibull model is 2.617, again very similar to the Cox model. While clearly not an exhaustive test, this example does demonstrate how similar the predictions given by each model are.

Given this, we might prefer the Weibull model over the Cox model in order to answer questions about our data set. The closed form hazard function gives a more complete description of our data, and in particular for the Weibull model can also be easily rephrased as an accelerated failure time model, letting us answer questions about expected goal times much more easily than the Cox model. While the flexibility of semi- or non-parametric models is often needed in situations with unusually behaved data, in our case these don't seem particularly necessary.

5.4 A brief look at a count model

Given a process with exponentially distributed inter-event times, the number of such events in some timespan will be Poisson distributed. Modelling the number of goals scored in a football match is often done through a Poisson approach, but methods have been developed to fit count models to data with Weibull distributed inter-event times.

Such models are often inexpressible in closed form, but series approximations exist for calculation. The probability of n events having occurred by time t can be expressed as the probability density of having the first event at time τ, multiplied by the probability of $n-1$ events having occured in the period between τ and t, integrated over τ. The first term is given by the pdf of the inter-event time distribution, $f(\tau)$, so this calculation gives a recursive solution for the count distribution, solved practically through Taylor series [24].

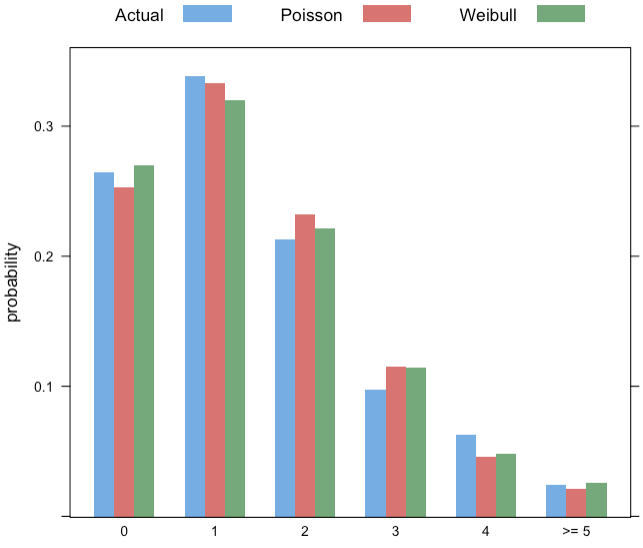


Figure 22: Bar plot of actual/fitted frequency for each goal count

Lucky for us, authors Kharrat and Boshnakov (mentioned in the literature review) have written the R package `countr`, which handles the model fitting. We will fit a Poisson and a Weibull renewal-count model, each against the same covariates as before.

In a different paper [25] the same authors show a plot similar to Figure 22, though ours differs in that the Weibull count model doesn't show any obvious improvement visually over the Poisson one. We can confirm our suspicions with a likelihood ratio test, which shows that with $\chi^2_{\text{df}=1} = 2.17, p = 0.141$ the extra shape parameter used in the Weibull count model doesn't significantly improve the fit. Our earlier analysis of first goals came to the opposite conclusion, possibly suggesting that the nature of 2nd, 3rd goals might be different from first goals. The authors mentioned examine over 1000 matches, so the issue could conceivably be one of sample size, though changing conditions in the sport could also be at play, as 7 years have passed since their paper. More work could be done to show why this difference in results has occured.

Bibliography

[1] Premier League, "Economic benefits of Premier League confirmed by report." [Online]. Available: https://www.premierleague.com/news/2434933

[2] Gambling Commission, "Annual Report and Accounts 2020 to 2021." [Online]. Available: https://www.gamblingcommission.gov.uk/report/annual-report-and-accounts-2020-to-2021/annual-report-20-21-performance-report-overview-of-the-british-gambling#ref-2

[3] Financial Times, "How English football became hooked on gambling." [Online]. Available: https://www.ft.com/content/9c3713ab-4317-4205-96ce-1075ecc5f865

[4] N. Razali, A. Mustapha, F. A. Yatim, and R. A. Aziz, "Predicting Football Matches Results using Bayesian Networks for English Premier League (EPL)," *IOP Conference Series: Materials Science and Engineering*, vol. 226, no. 1, p. 12099–12100, 2017, doi: 10.1088/1757-899X/226/1/012099.

[5] D. Y. X. C. Y. L. D. H Koo S. B Panday and H. Y. Kim, "Logistic Regression of Wins and Losses in Asia League Ice Hockey in the 2014-2015 Season," *International Journal of Performance Analysis in Sport*, vol. 16, no. 3, pp. 871–880, 2016, doi: 10.1080/24748668.2016.11868935.

[6] K. A. Willoughby, "Winning Games in Canadian Football: A Logistic Regression Analysis," *The College Mathematics Journal*, vol. 33, no. 3, pp. 215–220, 2002, doi: 10.1080/07468342.2002.11921944.

[7] D. Prasetio and D. Harlili, "Predicting football match results with logistic regression," in *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, 2016, pp. 1–5. doi: 10.1109/ICAICTA.2016.7803111.

[8] S. Ibáñez, J. Pérez-Goye, J. Courel Ibáñez, and J. García, "The impact of scoring first on match outcome in women's professional football," *International Journal of Performance Analysis in Sport*, vol. 18, pp. 1–9, 2018, doi: 10.1080/24748668.2018.1475197.

[9] D. Nevo and Y. Ritov, "Around the goal: examining the effect of the first goal on the second goal in soccer using survival analysis methods," *Journal of Quantitative Analysis in Sports*, vol. 9, no. 2, pp. 165–177, 2013, doi: doi:10.1515/jqas-2012-0004.

[10] M. J. Dixon and M. E. Robinson, "A Birth Process Model for Association Football Matches," *Journal of the Royal Statistical Society. Series D (The Statistician)*, vol. 47, no. 3, pp. 523–538, 1998, Accessed: Mar. 05, 2024. [Online]. Available: http://www.jstor.org/stable/2988632

[11] A. Adam, "Generalised Linear Model for Football Matches Prediction," in *MLSA@PKDD/ECML*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:13029746

[12] T. Kharrat and G. N. Boshnakov, "Football data analysis," 2019, Accessed: Mar. 05, 2024. [Online]. Available: https://cran-admin.ma.ic.ac.uk/web/packages/Countr/vignettes/exampleFootball.pdf

[13] J. J.P., "The home field advantage in athletics: A meta-analysis," *Journal of Applied Social Psychology*, vol. 40, no. 7, pp. 1819–1848, 2010, Accessed: Mar. 05, 2024. [Online]. Available: http://www.jstor.org/stable/2348899

[14] R. Pollard and V. Armatas, "Factors affecting home advantage in football World Cup qualification," *International Journal of Performance Analysis in Sport*, vol. 17, no. 1–2, pp. 121–135, 2017, doi: 10.1080/24748668.2017.1304031.

[15] S. R. Clarke and J. M. Norman, "Home Ground Advantage of Individual Clubs in English Soccer," *Journal of the Royal Statistical Society. Series D (The Statistician)*, vol. 44, no. 4, pp. 509–521, 1995, Accessed: Mar. 05, 2024. [Online]. Available: http://www.jstor.org/stable/2348899

[16] The Mirror, "Full details on every Premier League derby match during the 2022-23 season." [Online]. Available: https://www.mirror.co.uk/sport/football/news/premier-league-derby-fixtures-dates-27249622

[17] R. T. Stefani, "Observed Betting Tendencies and Suggested Betting Strategies for European Football Pools," *Journal of the Royal Statistical Society Series D: The Statistician*, vol. 32, no. 3, pp. 319–329, 1983, doi: 10.2307/2987938.

[18] D. Lee, "A comparison of choice-based landscape preference models between British and Korean visitors to National Parks," *Life Science Journal*, vol. 10, pp. 2028–2036, 2013.

[19] UCLA, "ORDINAL LOGISTIC REGRESSION | R DATA ANALYSIS EXAMPLES." [Online]. Available: https://stats.oarc.ucla.edu/r/dae/ordinal-logistic-regression/

[20] R. Brant, "Assessing Proportionality in the Proportional Odds Model for Ordinal Logistic Regression," *Biometrics*, vol. 46, no. 4, pp. 1171–1178, 1990, Accessed: Mar. 29, 2024. [Online]. Available: http://www.jstor.org/stable/2532457

[21] "Cox–Snell residuals." [Online]. Available: https://www.oxfordreference.com/view/10.1093/oi/authority.20110803095644940

[22] S. Park and D. J. Hendry, "Reassessing Schoenfeld Residual Tests of Proportional Hazards in Political Science Event History Analyses," *American Journal of Political Science*, vol. 59, no. 4, pp. 1072–1087, 2015, Accessed: Apr. 05, 2024. [Online]. Available: http://www.jstor.org/stable/24582966

[23] P. M. GRAMBSCH and T. M. THERNEAU, "Proportional hazards tests and diagnostics based on weighted residuals," *Biometrika*, vol. 81, no. 3, pp. 515–526, 1994, doi: 10.1093/biomet/81.3.515.

[24] B. McShane, M. Adrian, E. T. Bradlow, and P. S. Fader, "Count Models Based on Weibull Interarrival Times," *Journal of Business & Economic Statistics*, vol. 26, no. 3, pp. 369–378, Jul. 2008, doi: 10.1198/073500107000000278.

[25] G. Boshnakov, T. Kharrat, and I. G. McHale, "A bivariate Weibull count model for forecasting association football scores," *International Journal of Forecasting*, vol. 33, no. 2, pp. 458–466, 2017, doi: https://doi.org/10.1016/j.ijforecast.2016.11.006.

[26] R. H. Koning, "Balance in Competition in Dutch Soccer," *Journal of the Royal Statistical Society. Series D (The Statistician)*, vol. 49, no. 3, pp. 419–431, 2000, Accessed: Mar. 19, 2024. [Online]. Available: http://www.jstor.org/stable/2681066

[27] B. Torsney, "Fitting Bradley Terry Models Using a Multiplicative Algorithm," in *COMPSTAT 2004 — Proceedings in Computational Statistics*, J. Antoch, Ed., Heidelberg: Physica-Verlag HD, 2004, pp. 513–526.

[28] R. R. Davidson, "On Extending the Bradley-Terry Model to Accommodate Ties in Paired Comparison Experiments," *Journal of the American Statistical Association*, vol. 65, no. 329, pp. 317–328, 1970, Accessed: Mar. 20, 2024. [Online]. Available: http://www.jstor.org/stable/2283595

[29] R. R. Davidson and R. J. Beaver, "On Extending the Bradley-Terry Model to Incorporate Within-Pair Order Effects," *Biometrics*, vol. 33, no. 4, pp. 693–702, 1977, Accessed: Mar. 21, 2024. [Online]. Available: http://www.jstor.org/stable/2529467

Appendices

Blank/commented lines have been automatically removed from each script. This makes it a bit harder to read but saves about 20 pages.

Appendix 1: Weibull model selection

R output for model selection in survival analysis section. `hyptester` is a custom function for easier to read significance tests.

```
> # __ Full Parametric Model ___
> library(survival)
> library(flexsurv)
> hyptester <- function(model) {
+   # 90% significance tests on model coefficients
+   # Convenience function, easier to read this than the confidence intervals in the raw output
+   coeffs <- model$coefficients
+   ses <- sqrt(diag(vcov(model)))
+   zcrit <- qnorm(0.95)
+   upper <- coeffs + zcrit*ses
+   lower <- coeffs - zcrit*ses
+
+   for(i in 1:length(coeffs)) {
+     varname <- names(coeffs)[i]
+     if(varname == "rate" || varname == "shape" || varname == "scale") {
+       next
+     }
+     verdict <- ifelse(sign(lower[i]) != sign(upper[i]), "Reject", "Accept")
+     print(paste0("___", varname, "___"))
+
+     print(paste0("Lower:  ", format(round(lower[i], 2), nsmall = 3),", Upper:", format(round(upper[i], 2), nsmall = 3)))
+
+     print(paste0(verdict))
+   }
+ }
> valid_matches <- subset(match_data, !is.na(points2021) & !is.na(opponent_points2021))
> red_card_home <- ifelse(valid_matches$red_cards_home > 0, 1, 0)
> red_card_away <- ifelse(valid_matches$red_cards_away > 0, 1, 0)
> s <- Surv(valid_matches$time_to_first_goal_mins, valid_matches$first_goal_censored)
> model_wei<- flexsurvreg(formula = s ~ home_or_away*(red_card_home + red_card_away + distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
dist = "WeibullPH")
> model_exp<- flexsurvreg(formula = s ~ home_or_away*(red_card_home + red_card_away + distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
dist = "exponential")
> model_wei
Call:
flexsurvreg(formula = s ~ home_or_away * (red_card_home + red_card_away +
    distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
    dist = "WeibullPH")

Estimates:
                                       data mean  est       L95%      U95%      se        exp(est)  L95%      U95%
shape                                        NA    1.14673   1.05272   1.24913   0.05004        NA        NA        NA
scale                                        NA    0.00668   0.00341   0.01308   0.00229        NA        NA        NA
home_or_awayHome                          0.50000    0.06501  -0.21283   0.34286   0.14176   1.06717   0.80829   1.40896
red_card_home                             0.05515    0.44680  -0.14194   1.03553   0.30038   1.56329   0.86767   2.81660
red_card_away                             0.02941   -0.17849  -1.08734   0.73037   0.46371   0.83654   0.33711   2.07585
distance_groupingfarther                  0.50735   -0.45472  -0.74948  -0.15996   0.15039   0.63463   0.47261   0.85218
points2021                               57.17647    0.01063   0.00492   0.01633   0.00291   1.01068   1.00493   1.01646
opponent_points2021                      57.17647   -0.00718  -0.01328  -0.00109   0.00311   0.99284   0.98681   0.99891
home_or_awayHome:red_card_home            0.02757   -0.59405  -1.44307   0.25497   0.43318   0.55209   0.23620   1.29043
home_or_awayHome:red_card_away            0.01471    0.42517  -0.76498   1.61532   0.60723   1.52985   0.46534   5.02951
home_or_awayHome:distance_groupingfarther 0.25368    0.55401   0.15335   0.95467   0.20442   1.74022   1.16573   2.59782

N = 544,  Events: 400,  Censored: 144
Total time at risk: 27954
Log-likelihood = -2076.251, df = 11
AIC = 4174.503

> model_exp
Call:
flexsurvreg(formula = s ~ home_or_away * (red_card_home + red_card_away +
    distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
    dist = "exponential")

Estimates:
                                       data mean  est        L95%       U95%       se         exp(est)   L95%       U95%
rate                                         NA    0.012447   0.007336   0.021116   0.003357         NA         NA         NA
home_or_awayHome                          0.500000   0.057058  -0.220866   0.334981   0.141800   1.058717   0.801824   1.397914
red_card_home                             0.055147   0.427501  -0.160937   1.015939   0.300229   1.533421   0.851346   2.761956
red_card_away                             0.029412  -0.169362  -1.077349   0.738625   0.463267   0.844203   0.340497   2.093056
distance_groupingfarther                  0.507353  -0.436213  -0.730564  -0.141862   0.150182   0.646480   0.481637   0.867741
points2021                               57.176471   0.009900   0.004206   0.015594   0.002905   1.009949   1.004215   1.015716
opponent_points2021                      57.176471  -0.006667  -0.012747  -0.000586   0.003102   0.993356   0.987334   0.999414
home_or_awayHome:red_card_home            0.027574  -0.562738  -1.411000   0.285524   0.432795   0.569647   0.243899   1.330459
home_or_awayHome:red_card_away            0.014706   0.392136  -0.796580   1.580853   0.606499   1.480140   0.450868   4.859098
home_or_awayHome:distance_groupingfarther 0.253676   0.530829   0.130672   0.930986   0.204165   1.700341   1.139594   2.537009

N = 544,  Events: 400,  Censored: 144
Total time at risk: 27954
```

```
Log-likelihood = -2080.911, df = 10
AIC = 4181.822

> hyptester(model_wei)
[1] "___home_or_awayHome___"
[1] "Lower:  -0.170, Upper:0.300"
[1] "Reject"
[1] "___red_card_home___"
[1] "Lower:  -0.050, Upper:0.940"
[1] "Reject"
[1] "___red_card_away___"
[1] "Lower:  -0.940, Upper:0.580"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.700, Upper:-0.210"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.010, Upper:0.020"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___home_or_awayHome:red_card_home___"
[1] "Lower:  -1.310, Upper:0.120"
[1] "Reject"
[1] "___home_or_awayHome:red_card_away___"
[1] "Lower:  -0.570, Upper:1.420"
[1] "Reject"
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.220, Upper:0.890"
[1] "Accept"
> hyptester(model_exp)
[1] "___home_or_awayHome___"
[1] "Lower:  -0.180, Upper:0.290"
[1] "Reject"
[1] "___red_card_home___"
[1] "Lower:  -0.070, Upper:0.920"
[1] "Reject"
[1] "___red_card_away___"
[1] "Lower:  -0.930, Upper:0.590"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.680, Upper:-0.190"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.010, Upper:0.010"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___home_or_awayHome:red_card_home___"
[1] "Lower:  -1.270, Upper:0.150"
[1] "Reject"
[1] "___home_or_awayHome:red_card_away___"
[1] "Lower:  -0.610, Upper:1.390"
[1] "Reject"
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.200, Upper:0.870"
[1] "Accept"
> rm(model_exp, model_wei)
> #Mostly just removing the insignificant interaction terms, checking if they're significant on their own before removal.
> model_wei2<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021 + red_card_home + red_card_away, data = valid_matches,
dist = "WeibullPH")
> model_exp2<- flexsurvreg(formula =s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021 + red_card_home + red_card_away, data = valid_matches,
dist = "exponential")
> hyptester(model_wei2)
[1] "___home_or_awayHome___"
[1] "Lower:  -0.200, Upper:0.260"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.710, Upper:-0.220"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.010, Upper:0.020"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___red_card_home___"
[1] "Lower:  -0.230, Upper:0.480"
[1] "Reject"
[1] "___red_card_away___"
[1] "Lower:  -0.450, Upper:0.540"
[1] "Reject"
```

```
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.240, Upper:0.910"
[1] "Accept"
> hyptester(model_exp2)
[1] "___home_or_awayHome___"
[1] "Lower:  -0.200, Upper:0.260"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.690, Upper:-0.200"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.000, Upper:0.010"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___red_card_home___"
[1] "Lower:  -0.230, Upper:0.480"
[1] "Reject"
[1] "___red_card_away___"
[1] "Lower:  -0.450, Upper:0.540"
[1] "Reject"
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.220, Upper:0.880"
[1] "Accept"
> rm( model_exp2 , model_wei2)
> model_wei3<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021, data = valid_matches, dist = "WeibullPH")
> model_exp3<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021, data = valid_matches, dist = "exponential")
> hyptester(model_wei3)#double checking everything's significant
[1] "___home_or_awayHome___"
[1] "Lower:  -0.190, Upper:0.270"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.700, Upper:-0.220"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.010, Upper:0.020"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.240, Upper:0.900"
[1] "Accept"
> hyptester(model_exp3)
[1] "___home_or_awayHome___"
[1] "Lower:  -0.200, Upper:0.260"
[1] "Reject"
[1] "___distance_groupingfarther___"
[1] "Lower:  -0.690, Upper:-0.200"
[1] "Accept"
[1] "___points2021___"
[1] "Lower:  0.000, Upper:0.010"
[1] "Accept"
[1] "___opponent_points2021___"
[1] "Lower:  -0.010, Upper:0.000"
[1] "Accept"
[1] "___home_or_awayHome:distance_groupingfarther___"
[1] "Lower:  0.220, Upper:0.880"
[1] "Accept"
> model_exp3
Call:
flexsurvreg(formula = s ~ home_or_away * (distance_grouping) +
    points2021 + opponent_points2021, data = valid_matches, dist = "exponential")

Estimates:
                                       data mean  est        L95%       U95%       se         exp(est)   L95%       U95%
rate                                          NA   0.012849   0.007589   0.021755   0.003452         NA         NA         NA
home_or_awayHome                        0.500000   0.029845  -0.243987   0.303676   0.139713   1.030294   0.783498   1.354830
distance_groupingfarther                0.507353  -0.441736  -0.732257  -0.151215   0.148228   0.642919   0.480822   0.859663
points2021                             57.176471   0.009765   0.004019   0.015512   0.002932   1.009813   1.004027   1.015632
opponent_points2021                    57.176471  -0.006727  -0.012831  -0.000624   0.003114   0.993295   0.987251   0.999376
home_or_awayHome:distance_groupingfarther   0.253676   0.547532   0.152346   0.942717   0.201629   1.728980   1.164563   2.566947

N = 544,  Events: 400,  Censored: 144
Total time at risk: 27954
Log-likelihood = -2082.065, df = 6
AIC = 4176.13

> model_wei3
Call:
flexsurvreg(formula = s ~ home_or_away * (distance_grouping) +
    points2021 + opponent_points2021, data = valid_matches, dist = "WeibullPH")
```

```
Estimates:
                                       data mean  est       L95%      U95%      se        exp(est)  L95%      U95%
shape                                        NA    1.14469   1.05078   1.24698   0.04999        NA        NA        NA
scale                                        NA    0.00696   0.00356   0.01361   0.00238        NA        NA        NA
home_or_awayHome                          0.50000  0.03662  -0.23725   0.31049   0.13973   1.03730   0.78880   1.36410
distance_groupingfarther                  0.50735 -0.45986  -0.75064  -0.16909   0.14836   0.63137   0.47207   0.84443
points2021                               57.17647  0.01047   0.00472   0.01622   0.00294   1.01053   1.00473   1.01636
opponent_points2021                      57.17647 -0.00724  -0.01335  -0.00113   0.00312   0.99279   0.98674   0.99887
home_or_awayHome:distance_groupingfarther  0.25368  0.57123   0.17569   0.96677   0.20181   1.77045   1.19207   2.62943

N = 544,  Events: 400,  Censored: 144
Total time at risk: 27954
Log-likelihood = -2077.529, df = 7
AIC = 4169.059
```

Appendix 2: Data Prep

Code for data preparation.

```
data_path = "/Users/ralphbraithwaite/Documents/dissertation/data"
setwd(data_path)
rm(data_path)
locations <- read.csv("longlatpairs.csv")
prev_standings <- read.csv("prev_season_standings.csv")
footystats <- read.csv("footystats-match.csv")
footballdata <- read.csv("./Supplementary:Unused/footballdata-by-match.csv")
library(geosphere)
coordinates <- locations[, c("HomeLong", "HomeLat")]
pairwise_distances <- matrix(NA, nrow = nrow(locations), ncol = nrow(locations))
rownames(pairwise_distances) <- locations$Team
colnames(pairwise_distances) <- locations$Team
for (i in 1:nrow(locations)) {
  for (j in 1:nrow(locations)) {
    pairwise_distances[i, j] <- distVincentyEllipsoid(coordinates[i, ],
                                                      coordinates[j, ])
  }
}
pairwise_distances <- round(pairwise_distances/ 1000)
rm(coordinates, i, j, locations)
team_names <- sort(unique(footystats$home_team_name))
prev_season_data <- prev_standings[,c("Team", "PTS", "W", "Diff")]
data_by_team <- data.frame(team_names)
data_by_team <- merge(data_by_team,
                      prev_season_data,
                      by.x ="team_names",
                      by.y = "Team", all.x = TRUE)
colnames(data_by_team) <- c("team", "points", "wins", "goal_difference")
rm(prev_standings, team_names, prev_season_data)
find_first_goal_time <- function (timings) {
  if (timings == "") {
    is_censored = 0
    goal_time = 90
    return(c(goal_time, is_censored))
  }
  first_time = strsplit(timings,",")[[1]][1] #get first number preceding any comma, just the number if no comma
  extra_time_removed = strsplit(first_time, "'")[[1]][1] #if goal time is 45'2, 90'4 etc, just take the 45 or 90
  goal_time = as.integer(extra_time_removed)
  is_censored = 1
  if (goal_time == 0) {
    goal_time = 0.5
  }
  return(c(goal_time, is_censored))
}
time_to_first_goal_home <- t(sapply(footystats$home_team_goal_timings,
                                    find_first_goal_time))
time_to_first_goal_away <- t(sapply(footystats$away_team_goal_timings,
                                    find_first_goal_time))
match_distances <- pairwise_distances[cbind(footystats$home_team_name,
                                            footystats$away_team_name)]
home_win_margin <- footystats$home_team_goal_count - footystats$away_team_goal_count
result_for_home <- sign(home_win_margin)
result_for_away <- -sign(home_win_margin)
result_for_home <- factor(result_for_home, labels = c("Loss", "Draw", "Win"))
result_for_away <- factor(result_for_away, labels = c("Loss", "Draw", "Win"))
home_matches <- data.frame(
  footystats$timestamp,
  footystats$home_team_name,
  footystats$away_team_name,
  result_for_home,
  home_win_margin,
  footystats$home_team_goal_count,
  time_to_first_goal_home,
  match_distances,
  footystats$home_team_red_cards,
  footystats$away_team_red_cards,
  rep("Home",nrow(footystats))
```

```
)
away_matches <- data.frame(
  footystats$timestamp,
  footystats$away_team_name,
  footystats$home_team_name,
  result_for_away,
  home_win_margin,
  footystats$away_team_goal_count,
  time_to_first_goal_away,
  match_distances,
  footystats$home_team_red_cards,
  footystats$away_team_red_cards,
  rep("Away",nrow(footystats))
)
unique_matches <- data.frame(
  footystats$timestamp,
  footystats$home_team_name,
  footystats$away_team_name,
  result_for_home,
  home_win_margin,
  time_to_first_goal_home,
  match_distances,
  footystats$home_team_red_cards,
  footystats$away_team_red_cards,
  footballdata$B365H,
  footballdata$B365D,
  footballdata$B365A
)
rm(footystats, result_for_away, result_for_home, home_win_margin,
   time_to_first_goal_away, time_to_first_goal_home, match_distances,
   find_first_goal_time, pairwise_distances, footballdata)
cnames <- c("timestamp_unix", "team", "opponent",
            "result", "win_margin", "goal count", "time_to_first_goal_mins",
            "first_goal_censored", "away_distance_km",
            "red_cards_home", "red_cards_away", "home_or_away")
colnames(home_matches) <- c(cnames)
colnames(away_matches) <- c(cnames)
colnames(unique_matches) <- c("timestamp_unix", "team", "opponent",
                              "result", "win_margin", "time_to_first_goal_mins",
                              "first_goal_censored", "away_distance_km",
                              "red_cards_home", "red_cards_away", "bet365odds_home","bet365odds_draw", "bet365odds_away")
rownames(home_matches) <- NULL
rownames(away_matches) <- NULL
rownames(unique_matches) <- NULL
data_by_match <- rbind(home_matches, away_matches)
rm(cnames, home_matches, away_matches)
data_by_match <- merge(data_by_match,
                       data_by_team[,c("team", "points", "wins", "goal_difference")],
                       by.x = "team",
                       by.y = "team")
unique_matches <- merge(unique_matches,
                       data_by_team[,c("team", "points", "wins","goal_difference")],
                       by.x = "team",
                       by.y = "team")
is_derby <- function(teams) {
  rival_pairs = data.frame(
    # Could do this by distance instead
    #https://www.mirror.co.uk/sport/football/news/premier-league-derby-fixtures-dates-27249622
    # Used a small python script to parse these from the website
    A = c("Arsenal", "Tottenham Hotspur"),
    B = c("Leicester City", "Nottingham Forest"),
    C = c("Manchester City", "Manchester United"),
    D = c("West Ham United", "Fulham"),
    E = c("Brentford", "Chelsea"),
    F = c("Chelsea", "Arsenal"),
    G = c("West Ham United", "Crystal Palace"),
    H = c("Arsenal", "West Ham United"),
    I = c("Brentford", "Tottenham Hotspur"),
    J = c("West Ham United", "Brentford"),
    K = c("Aston Villa", "Wolves"),
    L = c("Crystal Palace", "Tottenham Hotspur"),
    M = c("Chelsea", "Crystal Palace"),
    N = c("Fulham", "Tottenham Hotspur"),
    O = c("Chelsea", "Fulham"),
    P = c("Crystal Palace", "Brighton & Hove Albion"),
    Q = c("Liverpool", "Everton"),
    R = c("West Ham United", "Chelsea"),
    S = c("Brentford", "Crystal Palace"),
    T = c("Tottenham Hotspur", "West Ham United"),
    U = c("Tottenham Hotspur", "Chelsea"),
    V = c("Brentford", "Fulham"),
    W = c("Fulham", "Arsenal"),
    X = c("Arsenal", "Crystal Palace")
  )
```

```
  any(sapply(rival_pairs, function(pair) { all(teams == pair) || all(rev(teams) == pair) }))
}
data_by_match <- cbind(
  data_by_match,
  apply(data_by_match[,c("team","opponent")],1,is_derby)
)
unique_matches <- cbind(
  unique_matches,
  apply(unique_matches[,c("team","opponent")],1,is_derby)
)
colnames(data_by_match)[ncol(data_by_match)] <- "match_is_derby"
colnames(unique_matches)[ncol(unique_matches)] <- "match_is_derby"
rm(is_derby)
breaks <- c(-Inf, median(data_by_team$points, na.rm=TRUE), Inf)
labels <- c("Lower", "Higher")
data_by_team$grouping <- cut(data_by_team$points, breaks = breaks, labels = labels, include.lowest = TRUE)
data_by_team$grouping[is.na(data_by_team$grouping)] <- "Lower"
rm(breaks)
add_grouping_columns <- function(this_df) {
  team_grouping = c()
  opponent_grouping = c()
  for (i in 1:(nrow(this_df))) {
    team_grouping[i] <- data_by_team$grouping[data_by_team$team == this_df$team[i]]
    opponent_grouping[i] <- data_by_team$grouping[data_by_team$team == this_df$opponent[i]]
  }

  team_grouping <- factor(team_grouping, levels = c(1,2), labels = labels)
  opponent_grouping <- factor(opponent_grouping, levels = c(1,2), labels = labels)
  prevnames <- names(this_df)

  this_df <- cbind(this_df, team_grouping, opponent_grouping)
  names(this_df) <- c(prevnames, "team_grouping", "opponent_grouping")
  return(this_df)
}
data_by_match <- add_grouping_columns(data_by_match)
unique_matches <- add_grouping_columns(unique_matches)
rm(add_grouping_columns, labels)
breaks = c( -Inf, median(data_by_match$away_distance_km), Inf)
data_by_match$distance_grouping <- cut(data_by_match$away_distance_km,
                                       breaks = c( -Inf, median(data_by_match$away_distance_km), Inf),
                                       labels = c("closer", "farther"),
                                       include.lowest = TRUE
)
unique_matches$distance_grouping <- cut(unique_matches$away_distance_km,
                                        breaks = c( -Inf, median(data_by_match$away_distance_km), Inf),
                                        labels = c("closer", "farther"),
                                        include.lowest = TRUE
)
rm(breaks)
data_by_match <- merge(data_by_match, data_by_team[c("team", "points")], by.x = "opponent", by.y = "team")
names(data_by_match)[names(data_by_match) == "points.x"] <- "points2021" # name got mangled by the merge, resetting
names(data_by_match)[names(data_by_match) == "points.y"] <- "opponent_points2021" # change name of new var
unique_matches <- merge(unique_matches, data_by_team[c("team", "points")], by.x = "opponent", by.y = "team")
names(unique_matches)[names(unique_matches) == "points.x"] <- "points2021"
names(unique_matches)[names(unique_matches) == "points.y"] <- "opponent_points2021"
write.csv(unique_matches, file = "./final/unique_redcards.csv")
write.csv(data_by_match, file = "./final/by_match_redcards.csv")
unique_matches <- subset(unique_matches, red_cards_home == 0 & red_cards_away == 0)
data_by_match <- subset(data_by_match, red_cards_home == 0 & red_cards_away == 0)
write.csv(data_by_team, file = "./final/by_team.csv")
write.csv(data_by_match, file = "./final/by_match.csv")
write.csv(unique_matches, file = "./final/unique.csv")
rm(unique_matches, data_by_match, data_by_team)
```

Appendix 3: Exploratory

Code for exploratory analysis.

```
setwd("/Users/ralphbraithwaite/Documents/dissertation/data/")
match_data <- read.csv("./final/by_match.csv")
team_data <- read.csv("./final/by_team.csv")
match_unique <- read.csv("./final/unique.csv")
colours = data.frame(
  blue = c("#BBD5F1", "#78ADE3", "#186FC6"),
  red = c("#EDBFBF","#D97676","#C62C2C"),
  green = c("#BCD4BE","#7AAB7C","#2F7833")
)
colours_transparent <- as.data.frame(lapply(colours, function(x) paste0(x, "7F")))
rownames(colours) <- c("pale", "mid", "vivid")
rownames(colours_transparent) <- c("pale", "mid", "vivid")
row_order <- c("Win", "Draw", "Loss")
result_table <- table(match_unique$result, match_unique$team_grouping)[row_order,]
ptab_result_group <- prop.table(result_table, margin = 2)
barplot(ptab_result_group, ylab = "Proportions of each result", xlab = "Team grouping based on last-season league points", main = "Results by grouping", col =
colours$blue, xlim = c(0,3))
legend("topright", legend = rev(rownames(ptab_result_group)), fill = rev(colours$blue))
```

```r
segments(x0 = 0, y0 = 0.5, x1 = 2.6, y1 = 0.5, lty = "dashed", col = "darkred")
text("50%", x= 2.6, y= 0.45, col = "darkred")
rm(result_table, ptab_result_group)
result_table <- table(match_data$result, match_data$team_grouping)[row_order,]
ptab_result_group <- prop.table(result_table, margin = 2)
barplot(ptab_result_group, ylab = "Proportions of each result", xlab = "Team grouping based on last-season league points", col = colours$red, xlim = c(0,3))
legend("topright", legend = rev(rownames(ptab_result_group)), fill = rev(colours$red))
segments(x0 = 0, y0 = 0.5, x1 = 2.6, y1 = 0.5, lty = "dashed", col = "darkred")
text("50%", x= 2.6, y= 0.45, col = "darkred")
rm(result_table, ptab_result_group)
tab_homeaway <- table(match_data$result, match_data$home_or_away)[row_order,]
ptab_homeaway <- prop.table(tab_homeaway, margin = 2)
barplot(ptab_homeaway, ylab = "Proportions of each result", xlab = "Home or Away games", col = colours$green, xlim = c(0,3))
legend("topright", legend = rev(rownames(ptab_homeaway)), fill = rev(colours$green))
segments(x0 = 0, y0 = 0.5, x1 = 2.6, y1 = 0.5, lty = "dashed", col = "darkred")
text("50%", x= 2.6, y= 0.45, col = "darkred")
rm(tab_homeaway, ptab_homeaway)
group_home_result_tab <- xtabs( data = match_data, formula = ~ result + home_or_away + team_grouping)
group_home_result_ptab <-  prop.table(group_home_result_tab)
two_var_stack_plotter <- function(ptable, this_title) {
  df <- data.frame(matrix(nrow = 3, ncol = 0)) # nrow needs to = 3 for cbind to work
  these_names <- c()
  for (i in c(1, 2)) {
    for (j in c(1, 2)) {
      newcol <- ptable[,i,j] / sum(ptable[,i,j])
      df <- cbind(df, newcol)
      fact1 <- strsplit(dimnames(ptable)[2][[1]], " ")[j]
      fact2 <- strsplit(dimnames(ptable)[3][[1]], " ")[i]
      these_names[2*(j-1)+(i)] <- paste0(fact1, " + ",  fact2)
    }
  }

  df <- df[c("Win", "Draw", "Loss"),]
  colnames(df) <- these_names
  print(df)

  barplot(height = as.matrix(df), col = colours$blue,
          xlim = c(0,6), ylim = c(0,1.19),
          main = this_title, xlab = "(Team + Opponent) Rankings")
  legend("topright", legend = rev(rownames(df)), fill = rev(colours$blue))
  segments(x0 = 0, y0 = 0.5, x1 = 5, y1 = 0.5, lty = "dashed", col = "darkred")
  text("50%", x= 5, y= 0.45, col = "darkred")
}
two_var_stack_plotter(group_home_result_ptab, "The effect of side + team grouping on match result")
rm(group_home_result_tab,group_home_result_ptab)
rankingvranking_tab <- xtabs( data = match_unique, formula = ~ result + opponent_grouping + team_grouping)
rvr_ptab <-  prop.table(rankingvranking_tab)
two_var_stack_plotter(rvr_ptab, "Home result by grouping of each team")
rankingvranking_tab <- xtabs( data = match_data, formula = ~ result + opponent_grouping + team_grouping)
rvr_ptab <-  prop.table(rankingvranking_tab)
two_var_stack_plotter(rvr_ptab, "Result for team by ranking of opponent")
flatRes <- ifelse(match_data$Result == "Win", "Win", "Not Win")
flatRes_unique <- ifelse(match_unique$result == "Win", "Win", "Not Win")
result_by_homeaway <- table(match_data$result, match_data$home_or_away)[row_order,]
result_by_grouping <- table(match_unique$result,match_unique$team_grouping)[row_order,]
result_by_derby <- table(match_unique$result, match_unique$match_is_derby)
median_distance <- median(unique(match_unique$away_distance_km))
above_median_distance <- ifelse(match_data$away_distance_km > median_distance, TRUE, FALSE)
plot(match_data$result, match_data$distance_km)
rm(median_distance, above_median_distance,result_by_derby, result_by_grouping, result_by_homeaway)
cor.test(team_data$points2021,team_data$wins2021) # r = 0.984, p = 1e-12
cor.test(team_data$points2021, team_data$goal_difference2021) # r = 0.967, p = 2e-10
rm(additional_predictors)
avgPoints <- mean(match_unique$points2021, na.rm = TRUE)
mod_allpoints <- lm(data = match_unique,
                    formula = win_margin ~  I(points2021 - avgPoints)+ I(opponent_points2021 - avgPoints))
summary(mod_allpoints)
inter <- mod_allpoints$coefficients[1]
al1 <- mod_allpoints$coefficients[2]
al2 <- mod_allpoints$coefficients[3]
exampleTeams <- c("Man City", "Arsenal", "Average", "Leeds")
examplePoints <- c(93, 69, avgPoints, 38)
these_colours <- c("#186FC6", "#C62C2C", "black", "#2F7833")

plot(match_unique$points, match_unique$win_margin, xlab = "Last-season points of home team", ylab = "Home team's win margin", main = "Team strength against win
margin", col = colours["mid","red"])
csts <- inter - (al1+al2)*avgPoints + al2*examplePoints
for (i in 1:4) {
    abline(a = csts[i], b = al1, col = these_colours[i], lty = "dashed")
}
legend(x= 77, y = 8.5, legend = paste0("Vs. ", exampleTeams), fill = these_colours)
abline(h=0, col = "grey75")
mod_test = lm(data = match_unique,
              formula = win_margin ~  points2021+ opponent_points2021)
```

Appendix 4: Logistic Regression

Code for logistic regression.

```
setwd("/Users/ralphbraithwaite/Documents/dissertation/data/")
team_data <- read.csv("./final/by_team.csv")
unique <- read.csv("./final/unique_redcards.csv")
colours = data.frame(
  blue = c("#BBD5F1", "#78ADE3", "#186FC6"),
  red = c("#EDBFBF","#D97676","#C62C2C"),
  green = c("#BCD4BE","#7AAB7C","#2F7833")
)
colours_transparent <- as.data.frame(lapply(colours, function(x) paste0(x, "7F")))
rownames(colours) <- c("pale", "mid", "vivid")
rownames(colours_transparent) <- c("pale", "mid", "vivid")
valid_matches <- subset(unique, !is.na(points2021) & !is.na(opponent_points2021))
result_bin <- ifelse(valid_matches$result == "Win", 1, 0)
red_card_home <- ifelse(valid_matches$red_cards_home > 0, 1, 0)
red_card_away <- ifelse(valid_matches$red_cards_away > 0, 1, 0)
late_season <- ifelse(valid_matches$timestamp_unix > median(valid_matches$timestamp_unix), 1, 0)
logreg_full <- glm(data = valid_matches, family = binomial(link="logit"), formula = result_bin ~ red_card_away + red_card_home + distance_grouping +
opponent_points2021 + points2021 + match_is_derby + late_season)
summary(logreg_full)
logreg_2 <- glm(data = valid_matches, family = binomial(link="logit"), formula = result_bin ~ red_card_home + distance_grouping + opponent_points2021 + points2021
+ late_season)
summary(logreg_2)
logreg_interactions <- glm( data = valid_matches, family = binomial(link="logit"), formula = result_bin ~ red_card_home + distance_grouping + opponent_points2021
+ (points2021)*late_season)
summary(logreg_interactions) #no relevant interactions - moving on. (I did test other interactions)
logreg_3 <- glm(data = valid_matches, family = binomial(link="logit"), formula = result_bin ~ red_card_home + opponent_points2021 + points2021 + distance_grouping)
lr3_summ <- summary(logreg_3)
lr3_summ
lr3_summ$coefficients
rm(logreg_2, logreg_full, logreg_interactions)
upper <- exp(lr3_summ$coefficients[,1] + 1.96*lr3_summ$coefficients[,2])
lower <- exp(lr3_summ$coefficients[,1] - 1.96*lr3_summ$coefficients[,2])
normal_values <- qnorm(c(0.025, 0.5, 0.975))
table <- matrix(nrow = 5, ncol = 3)
for (i in 1:5) {
  for(j in 1:3){
    table[i,j] <- exp(lr3_summ$coefficients[i,1] + normal_values[j]*lr3_summ$coefficients[i,2])
  }
}
table
prob_of_win <- function(homepoints, awaypoints, distance_level) {
  res <- predict(logreg_3,
          newdata = data.frame(red_card_home = 0, points2021 = homepoints, opponent_points2021 = awaypoints, distance_grouping = distance_level),
          type = "response")
  return( res )
}
x <- seq(35, 95, length.out = 100)
y <- seq(35, 95, length.out = 100)
grid <- expand.grid(x = x, y = y)
z <- prob_of_win(grid$x, grid$y, "farther")
filled.contour(x = x, y = y, z = matrix(z, nrow = length(x)),
               main = "Home win chance for more distant teams", xlab = "Home points", ylab = "Away Points",
              key.title = title(main = "Chance\nof win"),
              plot.axes = {
                axis(1)
                axis(2)
                contour(x,y,matrix(z, nrow = length(x)), add = TRUE, lwd = 2, levels = c(0.25,0.5,0.75) )
                })
segments(x0 = 30, y0 = 31.5, x1 = 81.65, y1 = 95.6, col = "black", lty = "dashed")
z <- prob_of_win(grid$x, grid$y, "closer")
filled.contour(x = x, y = y, z = matrix(z, nrow = length(x)),
               main = "Home win chance for closer located teams", xlab = "Home points", ylab = "Away Points",
               key.title = title(main = "Chance\nof win"),
               plot.axes = {
                 axis(1)
                 axis(2)
                 contour(x,y,matrix(z, nrow = length(x)), add = TRUE, lwd = 2, levels = c(0.25,0.5,0.75) )
               })
segments(x0 = 30, y0 = 31.5, x1 = 81.65, y1 = 95.6, col = "black", lty = "dashed")
model_null <- glm(result_bin ~ 1, family = binomial)
R2_MF <- 1-0.9210946
library(MASS)
results <- factor(valid_matches$result, labels = c("Loss", "Draw", "Win"))
olog_model_prelim <- polr(formula = results ~ red_card_home + red_card_away + points2021 + opponent_points2021 + distance_grouping + late_season, data =
valid_matches, Hess = TRUE) #all variables - too many low t values
summary(olog_model_prelim)
olog_model_2 <- polr(formula = results ~ red_card_home + points2021 + opponent_points2021 + distance_grouping, data = valid_matches, method="logistic")
summary(olog_model_2)
predict(predict(olog_model_2, newdata = data.frame(red_card_home = 0, points2021 = 74, opponent_points2021 = 69, distance_grouping = "closer"), type = "probs"))
library(nnet)
multinom_mod <- multinom( result ~ red_card_home + points2021 + opponent_points2021 + distance_grouping, data = valid_matches)
summary(multinom_mod)
```

```
df_olrm <- 12
df_multinom <- 20
p_val <- 1-pchisq(abs(olog_model_2$deviance-multinom_mod$deviance),   abs(df_multinom-df_olrm))
library(brant)
brant(olog_model_2)
olog_model_factors <- polr(formula = results ~ red_card_home + team_grouping+ opponent_grouping + distance_grouping, data = valid_matches, method="logistic")
brant(olog_model_factors) # this works - explain why.
set.seed(4)
sample <- sample(c(TRUE, FALSE), nrow(valid_matches), replace=TRUE, prob=c(0.8,0.2))
training  <- valid_matches[sample, ]
testing   <- valid_matches[!sample, ]
train_res_bin <- ifelse(training$result == "Win", 1, 0)
test_res_bin <- ifelse(testing$result == "Win", 1, 0)
red_card_train <- ifelse(training$red_cards_home > 0, 1, 0)
red_card_test <- ifelse(testing$red_cards_home > 0, 1, 0)
bookies_odds <- data.frame(testing[,c("bet365odds_home", "bet365odds_draw", "bet365odds_away")])
colnames(bookies_odds) <- c("Home", "Tie", "Away")
logreg_test <- glm(data = training, family = binomial(link="logit"), formula = train_res_bin ~ red_card_train + opponent_points2021 + points2021 +
distance_grouping)
to_predict <- data.frame(red_card_train = red_card_test, opponent_points2021 = testing$opponent_points2021, points2021 = testing$points2021, distance_grouping
= testing$distance_grouping)
lps <- predict(logreg_test, newdata = to_predict)
probs <- 1/(1 + exp(-1*lps))
table(ifelse(sign(lps) == 1, "Pred win", "Pred tie/loss"), ifelse(test_res_bin == 1, "Actual win", "Tie/loss"))
bookie_mat <- matrix(0, nrow =2, ncol =2)
our_mat <- matrix(0, nrow =2, ncol =2)
for (i in 1:nrow(testing)) {
  res <- ifelse(test_res_bin[i] ==1, 2, 1)
  bookie <- ifelse(match(min(bookies_odds[i,]),bookies_odds[i,]) == 1, 2, 1) # 1 home 2 tie 3 away
  pred <- ifelse(probs[i] > 0.5, 2, 1)
  bookie_mat[bookie, res] <- bookie_mat[bookie, res] + 1
  our_mat[pred, res] <- our_mat[pred, res] + 1
}
bookie_mat
our_mat
olrm_res_train <- factor(training$result, labels = c("Loss", "Draw", "Win"))
olrm_test <- polr(formula = olrm_res_train ~ red_card_train + opponent_points2021 + points2021 + distance_grouping, data = training, method="logistic")
to_predict <- data.frame(red_card_train = red_card_test, opponent_points2021 = testing$opponent_points2021, points2021 = testing$points2021, distance_grouping
= testing$distance_grouping)
probs <- predict(olrm_test, newdata = to_predict,type = "probs")
get_min_column <- function(row) {
  names(which.min(row))[1]
}
get_max_column <- function(row) {
  names(which.max(row))[1]
}
bookie_pred <- c()
our_pred <- c()
for (i in 1:nrow(testing)) {
  bookie_pred <- append(bookie_pred, get_min_column(bookies_odds[i,]))
  our_pred <- append(our_pred, get_max_column(probs[i,]))
}
print(bookie_pred)
print(our_pred)
table(bookie_pred, testing$result)
table(our_pred, testing$result)
olrm_res_train <- factor(training$result, labels = c("Loss", "Draw", "Win"))
olrm_testfact <- polr(formula = olrm_res_train ~ red_card_train + team_grouping + opponent_grouping + distance_grouping, data = training, method="logistic")
to_predict <- data.frame(red_card_train = red_card_test, opponent_grouping = testing$opponent_grouping, team_grouping= testing$team_grouping, distance_grouping
= testing$distance_grouping)
probs <- predict(olrm_testfact, newdata = to_predict,type = "probs")
bookie_pred <- c()
our_pred <- c()
for (i in 1:nrow(testing)) {
  bookie_pred <- append(bookie_pred, get_min_column(bookies_odds[i,]))
  our_pred <- append(our_pred, get_max_column(probs[i,]))
}
print(bookie_pred)
print(our_pred)
table(bookie_pred, testing$result)
table(our_pred, testing$result)
```

Appendix 5: Survival Analysis

Code for survival analysis.

```
setwd("/Users/ralphbraithwaite/Documents/dissertation/data/")
match_data <- read.csv("./final/by_match_redcards.csv")
team_data <- read.csv("./final/by_team.csv")
match_unique <- read.csv("./final/unique_redcards.csv")
colours = data.frame(
  blue = c("#BBD5F1", "#78ADE3", "#186FC6"),
  red = c("#EDBFBF","#D97676","#C62C2C"),
  green = c("#BCD4BE","#7AAB7C","#2F7833")
)
```

```r
colours_transparent <- as.data.frame(lapply(colours, function(x) paste0(x, "7F")))
rownames(colours) <- c("pale", "mid", "vivid")
rownames(colours_transparent) <- c("pale", "mid", "vivid")
library(survival)
library(ggplot2)
one_line_plot<- function (survival_model, this_title) {
  # Give this function the output of a survfit() function
  quarts = c(0.25, 0.5, 0.75)
  quart_labs = paste0( as.character(100 * quarts), "%")
  decs = seq(0, 1, 0.1)
  quints = seq(0, 1, 0.2)
  y_labels = c( "0%", "", "20%", "", "40%", "", "60%", "", "80%", "", "100%")
  xticks = seq(0, 90, 15)
  xbounds = c(0, 90)

  summ <- summary(survival_model)
  plot(summ$time, summ$surv, main = this_title, ylab = "Percent chance of having scored no goal",
       xlab = "Match Time (mins)", axes = 0, ylim = c(0,1), xlim = c(0,90))
  # 95% Confidence region1
  polygon(c(summ$time, rev(summ$time)), c(summ$lower, rev(summ$upper)), col = "grey75", border = FALSE)
  lines(summ$time, summ$surv, type="s")

  axis(1, at = xticks, lab = xticks, pos = 0) # x axis
  axis(2, at = decs, lab = y_labels, pos = 0) # y axis
  axis(4, at = decs, lab = y_labels, pos = 90) # y axis - right side
  segments( x0 = 2, y0 = quarts, x1 = 88, y1 = quarts, lty = "dashed", col = "darkred") # Quartile markers
  text(quart_labs, x = 10, y= quarts - 0.05, col = "darkred")   # Quartile labels
  segments( x0 = 0, y0 = 1, x1 = 90, y1 = 1, col = "darkgrey") #plot boundary - top
  legend(x = 72, y = 0.99, legend= "95% CI", fill = "grey75")
}
two_line_plot<- function (surv_mod, this_title, reverse_order) {

  summ <- summary(surv_mod)
  levs <- levels(summ$strata)
  indic1 <- which(summ$strata == levs[1])
  indic2 <- which(summ$strata == levs[2])
  time1 <- summ$time[indic1]
  time2 <- summ$time[indic2]
  lower1 <- summ$lower[indic1]
  lower2 <- summ$lower[indic2]
  upper1 <- summ$upper[indic1]
  upper2 <- summ$upper[indic2]
  surv1 <- summ$surv[indic1]
  surv2 <- summ$surv[indic2]
  name1 <- strsplit(levs, "=")[[1]][2]
  name2 <- strsplit(levs, "=")[[2]][2]

  quarts = c(0.25, 0.5, 0.75)
  quart_labs = paste0( as.character(100 * quarts), "%")
  decs = seq(0, 1, 0.1)
  quints = seq(0, 1, 0.2)
  y_labels = c( "0%", "", "20%", "", "40%", "", "60%", "", "80%", "", "100%")
  xticks = seq(0, 90, 15)
  xbounds = c(0, 90)

  plot(surv_mod, main = this_title, ylab = "Percent chance of having scored no goal",
       xlab = "Match Time (mins)", axes = 0, ylim = c(0,1), xlim = c(0,90))
  fills = colours_transparent[c("blue","red")][1,]
  if (reverse_order) {
    polygon(c(time2, rev(time2)), c(lower2, rev(upper2)), col = fills$red, border = FALSE)
    polygon(c(time1, rev(time1)), c(lower1, rev(upper1)), col = fills$blue, border = FALSE)
    lines(time2, surv2, type="s", col = colours["mid", "red"])
    lines(time1, surv1, type="s", col = colours["mid", "blue"])
  }
  else {
    polygon(c(time1, rev(time1)), c(lower1, rev(upper1)), col = fills$blue, border = FALSE)
    polygon(c(time2, rev(time2)), c(lower2, rev(upper2)), col = fills$red, border = FALSE)
    lines(time1, surv1, type="s", col = colours["mid", "blue"])
    lines(time2, surv2, type="s", col = colours["mid", "red"])
  }
  axis(1, at = xticks, lab = xticks, pos = 0) # x axis
  axis(2, at = decs, lab = y_labels, pos = 0) # y axis
  axis(4, at = decs, lab = y_labels, pos = 90) # y axis - right side
  segments( x0 = 2, y0 = quarts, x1 = 88, y1 = quarts, lty = "dashed", col = "darkred") # Quartile markers
  text(quart_labs, x = 10, y= quarts - 0.05, col = "darkred")   # Quartile labels
  segments( x0 = 0, y0 = 1, x1 = 90, y1 = 1, col = "darkgrey") #plot boundary - top

  legend(x = 72, y = 0.99, legend = c(name2, name1), fill = c(colours["mid","red"], colours["mid", "blue"]))
}
cum_haz_plot<- function (survival_model, this_title) {
  # Give this function the output of a survfit() function
  xticks = seq(0, 90, 15)
  xbounds = c(0, 90)
  summ <- summary(survival_model)
```

```
  low <- -log(summ$lower)
  upp <- -log(summ$upper)
  ymax = ceiling(max(low)/0.2) * 0.2
  plot(summ$time, summ$cumhaz, main = this_title, ylab = "Cumulative Hazard",
       xlab = "Match Time (mins)", axes = 0, ylim = c(0, ymax), xlim = c(0,90))

  # 95% Confidence region
  low <- -log(summ$lower)
  upp <- -log(summ$upper)
  polygon(c(summ$time, rev(summ$time)), c(low, rev(upp)), col = "grey75", border = FALSE)

  lines(summ$time, summ$cumhaz, type="s") #actual plot

  # line of best fit
  l <- lm(summ$cumhaz ~ 0 + summ$time)
  abline(a = 0, b = l$coefficients, col = "darkred", lty = "dashed")

  print("Gradient of best fit line:")
  print(l$coefficients)

  yticks = seq(0,ymax,0.2)
  axis(2, at = yticks, lab = yticks, pos = 0) # y axis
  axis(4, at = yticks, lab = yticks, pos = 90) # y axis - right side
  axis(1, at = xticks, lab = xticks, pos = 0) # x axis
  legend(x = 5, y = ymax * 0.95, legend= c("95% CI", "Linear fit"), fill = c("grey75", "darkred"))
}
two_line_cumhaz<- function (surv_mod, this_title, flip_line) {

  summ <- summary(surv_mod)
  levs <- levels(summ$strata)
  indic1 <- which(summ$strata == levs[1])
  indic2 <- which(summ$strata == levs[2])
  time1 <- summ$time[indic1]
  time2 <- summ$time[indic2]
  cumhaz1 <- summ$cumhaz[indic1]
  cumhaz2 <- summ$cumhaz[indic2]
  name1 <- strsplit(levs, "=")[[1]][2]
  name2 <- strsplit(levs, "=")[[2]][2]
  ymax <- ceiling(max(cumhaz2, cumhaz1)*5)/5
  quints = seq(0, ymax, 0.2)
  xticks = seq(0, 90, 15)
  xbounds = c(0, 90)

  plot(summ$time, summ$cumhaz, main = this_title, ylab = "Cumulative Hazard",
       xlab = "Match Time (mins)", axes = 0, ylim = c(0,ymax), xlim = c(0,90), col = "white")
  fills = colours_transparent[c("blue","red")][1,]
  # Cum haz lines
  lines(time2, cumhaz2, type="s", col = colours["mid", "red"])
  lines(time1, cumhaz1, type="s", col = colours["mid", "blue"])
  # Log diff line
  interp_cumhaz1 <- approx(x = time1, y = cumhaz1, xout = time2)$y
  lines(time2, flip_line*(log(cumhaz2) - log(interp_cumhaz1)), type = "s", col = "black")
  # Best fit
  l1 <- lm(cumhaz1 ~ 0 + time1)
  l2 <- lm(cumhaz2 ~ 0 + time2)
  abline(a = 0, b = l1$coefficients, col = colours["vivid","blue"], lty = "dashed")
  abline(a = 0, b = l2$coefficients, col = colours["vivid","red"], lty = "dashed")
  abline(h = flip_line*(log(l2$coefficients) -log(l1$coefficients)), col = "darkgray", lty = "dashed")
  print(paste0(name1, "coefficient is ", l1$coefficients))
  print(paste0(name2, "coefficient is ", l2$coefficients))
  print(paste0("their log difference is ", flip_line*(log(l2$coefficients) -log(l1$coefficients))))
  print(paste0("mean of line is ", mean(flip_line*(log(cumhaz2) - log(interp_cumhaz1)))))
  axis(1, at = xticks, lab = xticks, pos = 0) # x axis
  axis(2, at = quints, lab = quints, pos = 0) # y axis
  axis(4, at = quints, lab = quints, pos = 90) # y axis - right side

  legend(x = 4, y = 1.3, legend= c(name1, name2, "Difference of logs"), fill = c(colours["mid","blue"], colours["mid", "red"], "black"))
}
four_line_plot<- function (surv_mod, this_title, rates, scale_par) {
  # rates must be sorted
  # works for exp or wei dists only
  summ <- summary(surv_mod)
  levs <- levels(summ$strata)
  indic1 <- which(summ$strata == levs[1])
  indic2 <- which(summ$strata == levs[2])
  indic3 <- which(summ$strata == levs[3])
  indic4 <- which(summ$strata == levs[4])
  time1 <- summ$time[indic1]
  time2 <- summ$time[indic2]
  time3 <- summ$time[indic3]
  time4 <- summ$time[indic4]

  surv1 <- summ$surv[indic1]
```

```
  surv2 <- summ$surv[indic2]
  surv3 <- summ$surv[indic3]
  surv4 <- summ$surv[indic4]

  quarts = c(0.25, 0.5, 0.75)
  quart_labs = paste0( as.character(100 * quarts), "%")
  decs = seq(0, 1, 0.1)
  quints = seq(0, 1, 0.2)
  y_labels = c( "0%", "", "20%", "", "40%", "", "60%", "", "80%", "", "100%")
  xticks = seq(0, 90, 15)
  xbounds = c(0, 90)

  plot(surv_mod, main = this_title, ylab = "Percent chance of having scored no goal",
       xlab = "Match Time (mins)", axes = 0, ylim = c(0,1), xlim = c(0,90))
  lines(time4, surv4, type="s", col = "orange")
  lines(time3, surv3, type="s", col = "blue")
  lines(time2, surv2, type="s", col = "green")
  lines(time1, surv1, type="s", col = "red")

  times <- sort(unique(summ$time))
  lines(times, exp(-(times/exp(rates[1]))^(1/scale_par)), col = "orange", lty = "dashed") #
  lines(times, exp(-(times/exp(rates[2]))^(1/scale_par)), col = "blue", lty = "dashed")
  lines(times, exp(-(times/exp(rates[3]))^(1/scale_par)), col = "green", lty = "dashed")
  lines( times, exp( -(times/exp(rates[4]))^(1/scale_par) ), col = "red", lty = "dashed")
  axis(1, at = xticks, lab = xticks, pos = 0) # x axis
  axis(2, at = decs, lab = y_labels, pos = 0) # y axis
  axis(4, at = decs, lab = y_labels, pos = 90) # y axis - right side
  segments( x0 = 2, y0 = quarts, x1 = 88, y1 = quarts, lty = "dashed", col = "darkred") # Quartile markers
  text(quart_labs, x = 10, y= quarts - 0.05, col = "darkred")   # Quartile labels
  segments( x0 = 0, y0 = 1, x1 = 90, y1 = 1, col = "darkgrey") #plot boundary - top
}
surv <- Surv(match_data$time_to_first_goal_mins, match_data$first_goal_censored)
surv_model <- survfit(
  surv ~ 1
)
one_line_plot(surv_model, "Survival function for the time to first goal")
cum_haz_plot(surv_model, "Cumulative hazard plot for first-goal times")
surv_homeaway <- survfit(
  surv ~ match_data$home_or_away
)
two_line_plot(surv_homeaway, "Time to first goal for home games vs away", 0)
two_line_cumhaz( surv_homeaway, "Cumulative hazard for home vs away sides", 1)
survdiff(surv ~ match_data$home_or_away, rho = 0)
plot(1, type="n", xlab="Log(Match Time)", ylab="Log(- CH)", xlim=c(0, 4.5), ylim=c(-4.554,0.4156), main = "Graphical test for Weibull PH, home vs away")
lines(log(surv_homeaway$time[1:84]),log(surv_homeaway$cumhaz[1:84]), col = colours$red[3])
lines(log(surv_homeaway$time[85:169]),log(surv_homeaway$cumhaz[85:169]), col = colours$blue[3])
surv_grouping <- survfit(
  surv ~  match_data$team_grouping
)
two_line_plot(surv_grouping, "Time to first goal by grouping", 0)
two_line_cumhaz( surv_grouping, "Cumulative hazard for higher vs lower ranked teams", -1)
survdiff(surv ~ match_data$team_grouping, rho = 0)
surv_derby <- survfit(
  surv ~ match_data$match_is_derby
)
two_line_plot(surv_derby, "Time to first goal for derby games (TRUE) vs non derby games", 1)
two_line_cumhaz( surv_derby, "Cumulative hazard for (non-/)derby games", -1)
survdiff(surv ~ match_data$match_is_derby, rho = 1)
away_games <- subset(match_data, home_or_away == "Away")
surv_distance <- survfit(
  Surv(away_games$time_to_first_goal_mins, away_games$first_goal_censored) ~ away_games$distance_grouping
)
two_line_plot(surv_distance, "Time to first goal in away games by distance travelled",0)
two_line_cumhaz( surv_distance, "Cumulative hazard by distance travelled for away sides", -1)
survdiff(Surv(away_games$time_to_first_goal_mins, away_games$first_goal_censored) ~ away_games$distance_grouping, rho = 1)
rm(surv_model, surv_homeaway, surv_grouping, surv_derby, surv_distance, away_games)
library(dplyr)
median_distances <- match_unique %>%
  group_by(opponent) %>%
  summarise(median_distance = median(away_distance_km, na.rm = TRUE))
print(median_distances)
cor.test(median_distances$median_distance, team_data$points)
rm(median_distances)
half_season <- ifelse(match_data$timestamp_unix> median(match_data$timestamp_unix), "Later", "Earlier")
surv_timestamp <- survfit(
  surv ~ half_season
)
two_line_plot(surv_timestamp, "Time to first goal for games earlier/later in the season", 1)
two_line_cumhaz( surv_timestamp, "Cumulative hazard", -1)
survdiff(surv ~ half_season, rho = 1)
two_line_plot(surv_redcards, "Time to first goal for games earlier/later in the season", 1)
two_line_cumhaz( surv_redcards, "Cumulative hazard", 1)
home_games <- subset(match_data, home_or_away == "Home")
survdiff(Surv(home_games$time_to_first_goal_mins, home_games$first_goal_censored) ~ ifelse(home_games$red_cards_home>0, 1, 0), rho = 1)
```

```r
surv_oppgrouping <- survfit( surv ~ opponent_grouping, data = match_data) #Worth studying
two_line_plot(surv_oppgrouping, "Time to first goal for matches by strength of opponent", 0)
two_line_cumhaz( surv_oppgrouping, "Cumulative hazard by grouping of opponent", 1)
survdiff(surv ~ match_data$opponent_grouping, rho = 0)
rm(half_season, surv_timestamp, surv_oppgrouping)
library(survival)
library(flexsurv)
hyptester <- function(model) {
  # 90% significance tests on model coefficients
  # Saves my eyes a bit, convenience function
  coeffs <- model$coefficients
  ses <- sqrt(diag(vcov(model)))
  zcrit <- qnorm(0.95)
  upper <- coeffs + zcrit*ses
  lower <- coeffs - zcrit*ses

  for(i in 1:length(coeffs)) {
    varname <- names(coeffs)[i]
    if(varname == "rate" || varname == "shape" || varname == "scale") {
      next
    }
    verdict <- ifelse(sign(lower[i]) != sign(upper[i]), "Reject", "Accept")
    print(paste0("___", varname, "___"))

    print(paste0("Lower:  ", format(round(lower[i], 2), nsmall = 3),", Upper:", format(round(upper[i], 2), nsmall = 3)))

    print(paste0(verdict))
  }
}
valid_matches <- subset(match_data, !is.na(points2021) & !is.na(opponent_points2021))
red_card_home <- ifelse(valid_matches$red_cards_home > 0, 1, 0)
red_card_away <- ifelse(valid_matches$red_cards_away > 0, 1, 0)
red_card_team <- c()
red_card_opp <- c()
for(i in 1:nrow(valid_matches)) {
  if(valid_matches$home_or_away[i] == "Home"){
    red_card_team[i] <- red_card_home[i]
    red_card_opp[i] <- red_card_away[i]
  } else {
    red_card_team[i] <- red_card_away[i]
    red_card_opp[i] <- red_card_home[i]
  }
}
rm(red_card_away, red_card_home)
s <- Surv(valid_matches$time_to_first_goal_mins, valid_matches$first_goal_censored)
model_wei<- flexsurvreg(formula = s ~ home_or_away*(red_card_team + red_card_opp + distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
dist = "WeibullPH")
model_exp<- flexsurvreg(formula = s ~ home_or_away*(red_card_team + red_card_opp + distance_grouping) + points2021 + opponent_points2021, data = valid_matches,
dist = "exponential")
model_wei
model_exp
hyptester(model_wei)
hyptester(model_exp)
rm(model_exp, model_wei)
model_wei2<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021 + red_card_home + red_card_away, data = valid_matches,
dist = "WeibullPH")
model_exp2<- flexsurvreg(formula =s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021 + red_card_home + red_card_away, data = valid_matches,
dist = "exponential")
hyptester(model_wei2)
hyptester(model_exp2)
rm( model_exp2 , model_wei2)
model_wei3<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021, data = valid_matches, dist = "WeibullPH")
model_exp3<- flexsurvreg(formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021, data = valid_matches, dist = "exponential")
hyptester(model_wei3)
hyptester(model_exp3) #double checking
model_exp3
model_wei3
cs_wc <- coxsnell_flexsurvreg(model_wei3)
cs_ec <- coxsnell_flexsurvreg(model_exp3)
cstest_wc <- survfit(Surv(cs_wc$est, valid_matches$first_goal_censored) ~ 1)
plot(cstest_wc, fun="cumhaz", main = "Weibull",ylim = c(0,4), xlim = c(0,2.5))
abline(0, 1, col="red")
cstest_ec <- survfit(Surv(cs_ec$est, valid_matches$first_goal_censored) ~ 1)
plot(cstest_ec, fun="cumhaz", main = "Exponential", ylim = c(0,4), xlim = c(0,2.5))
abline(0, 1, col="red")
chisq_obs <- -2*(model_exp3$loglik-model_wei3$loglik) # =9.07
1-pchisq(9.07, 1)
library(survminer)
coxreg <- coxph(data = valid_matches, formula = s ~ home_or_away*( distance_grouping) + points2021 + opponent_points2021)
summary(coxreg)
cox_test <- cox.zph(coxreg)
cox_test
plot(cox_test, main = "Cox Residual Plot", xlab = "Match time", xaxp  = c(0,90,15))
abline(h=0, col = "darkred", lty = "dashed")
```

```r
ggcoxzph_fixed(cox_test)
to_predict <- data.frame(home_or_away = c("Home", "Away"), points2021 = c(69, 49), opponent_points2021 = c(49,69), distance_grouping = c("farther", "farther"))
lps <- predict(coxreg, newdata=to_predict, type = "lp", reference = "zero")
hrs <- exp(lps)
lps
hrs # To double check the workings-out
hrs[1]/hrs[2]
hr_flexsurvreg(x = model_wei3, newdata = to_predict, t = seq(0,90,1))

rm(colours, colours_transparent, cox_test, coxreg, match_data, match_unique, model_ctns_final, team_data, actual_distance, surv, cum_haz_plot, four_line_plot,
one_line_plot, two_line_cumhaz, two_line_plot)
library(Countr)
table(match_data$goal.count, match_data$home_or_away)
count_poiss <- glm(formula = goal.count ~ home_or_away*distance_grouping+points2021+opponent_points2021, family = poisson, data = valid_matches)
count_wei <- renewalCount(formula = goal.count ~ home_or_away*distance_grouping+points2021+opponent_points2021, data = valid_matches, dist = "weibull")
predictions <- compareToGLM(poisson_model = count_poiss, weibull = count_wei, breaks = 0:5)
colnames(predictions) <- c("Counts", "Actual", "Weibull", "Poisson", "weibull_pearson", "poisson_pearson")
frequency_plot(count_labels = predictions$Counts, actual =  predictions$Actual, pred = predictions[c("Weibull", "Poisson")], colours = c("#78ADE3","#D97676",
"#7AAB7C"))
lmtest::lrtest(count_wei, count_poiss) # Weibull is no better
chiSq_gof(count_poiss,breaks = 0:5)
chiSq_gof(count_wei,breaks = 0:5) #doesn't work - wonder why
home_counts <- rep(0,90)
away_counts <- rep(0,90)
home_bins <- rep(0,18)
away_bins <- rep(0,18)
footystats <- read.csv("footystats-match.csv")
for (i in 1:nrow(footystats)){
  hometimes <- footystats$home_team_goal_timings[i]
  awaytimes <- footystats$away_team_goal_timings[i]
  home_goals_int <- strsplit(hometimes, ",")[[1]]
  away_goals_int <- strsplit(awaytimes, ",")[[1]]
  for (s in home_goals_int) {
    goaltime <- as.numeric(strsplit(s, "'")[[1]][1])
    goal_bin <- floor(goaltime/5)
    home_bins[goal_bin] <- home_bins[goal_bin] +1
    home_counts[goaltime] <- home_counts[goaltime] +1
  }
  for (s in away_goals_int) {
    goaltime <- as.numeric(strsplit(s, "'")[[1]][1])
    goal_bin <- floor(goaltime/5)
    away_bins[goal_bin] <- away_bins[goal_bin] +1
    away_counts[goaltime] <- away_counts[goaltime] +1
  }
}
barplot(home_bins, ylim = c(0,60)
axis(side=2, pos= -2, at = seq(0,60,15), ylim = c(0,60))
axis(side=1, pos = 0, at = seq(0,18,2))
s <- Surv(valid_matches$time_to_first_goal_mins, valid_matches$first_goal_censored)
valid_model <-survfit( s ~ 1 )
weib_line <- 1.14469*0.00696*(surv_model$time)^(0.14469)
one_line_plot(valid_model, "Kaplan Meier survival curve, Weibull fit overlaid")
lines(valid_model$time, weib_line)
lines(surv_model$time, exp(-0.00696*1.14*(surv_model$time)^(1.14469)))
vcens <- subset(valid_matches, first_goal_censored != 0)
hist(vcens$time_to_first_goal_mins)
lines(surv_model$time,1.14469*0.00696*(surv_model$time)^(0.14469))
lines(surv_model$time, 70*surv_model$surv)
```

Appendix 6: Bradley Terry models

Unfinished section on Bradley Terry models. Due to a different approach and a few errors this was written under the impression that the logistic regression portion of this thesis was much less fruitful.

Bradley-Terry models

In this section we will take a new approach to the problem of modelling football match results. A paper by R. H. Koning [26] uses an ordered probit model to assess team strengths and home advantage. Using D^* as a latent variable for match result, the author fits a model as $D^* = \alpha_i - \alpha_j + h_{ij} + \nu_{ij}$, where α_k is a term representing the relative strength of team k, h_{ij} represents a home advantage factor when i plays at home against j, and ν_{ij} are normal errors. The model is thus treating the outcome of a game as being related to the difference in "strength" between the two teams, with some extra detail through the error and home advantage terms.

This is very similar to a Bradley-Terry model (BT model), and in the following section we will see if these are useful for modelling match outcomes for our data. The BT model adapts logistic regression to treat the response variable as a difference of "strength" values: $\text{logit}(\Pr(i > j)) = \alpha_i - \alpha_j$ A more common use of regression in sports involves taking information about the strengths of each and using it to predict match outcome, whereas BT models use data on match outcome to assign strengths to teams. In addition to sports modelling, these models are often used for ranking items based on a series of pairwise comparisons. The usual BT model isn't equipped to deal with ties in the data so as in the logistic regression section, the outcomes here will be 'home win' and 'not home win'. An advantage of these models over the previous approach is that we aren't using any predictive variables, only match results against dummy variables for the teams, hence the low information content of our data won't hamper model performance. We can fit a basic model using `BTm()` in the `BradleyTerry2` package in R. For this section we are including every match result, including games involving promoted teams and matches with red cards.

```
bt_mod <- BTm(outcome = result_bin,
          player1 = factor(match_unique$team),
          player2 = factor(match_unique$opponent))
summary(bt_mod)
```

Which finds the following list of team specific "strengths".

Coefficients:	
Arsenal	1.964
Aston Villa	1.173
Brentford	1.054
Brighton & HA	1.054
Chelsea	0.242
Crystal Palace	0.359
Everton	0.476
Fulham	0.476
Leeds United	−0.392
Leicester City	−0.126
Liverpool	1.294
Manchester City	2.469
Manchester United	1.546
Newcastle United	1.679
Nottingham Forest	0.122
Southampton	−0.687
Tottenham Hotspur	1.173
West Ham United	0.122
Wolverhampton Wanderers	0.359

Notice the absence of AFC Bournemouth among the teams listed, they are treated as the reference team with strength 0. The values given seem to make some sense, with stronger teams such as Manchester City and Liverpool having higher coefficients and weaker teams such as Leeds having negative ones. All of the coefficients here have a standard error in the range $(0.49, 0.58)$. This is of course quite high, given many of the teams have coefficients of a magnitude lower than this. The high variance in these estimates is because each pairwise matchup occurs only twice, each team plays every other at home and away. Including multiple seasons worth of data may assuage this somewhat, as the results of more matchups could be included, however as teams move between leagues and their prowess changes over time this would add complications.

We can use the results of this fit to find predicted win chances for different teams. Taking Newcastle United (H) and Leicester City as examples:

$$\Pr(\text{N.U. Win}) = \frac{1}{1 + e^{-1.679+0.126}} = 0.825$$

Due to the fact that home teams win about half of their games and the outcomes are being grouped as "home win" vs "not home win", there's an argument against including a home advantage parameter here: the data support a roughly even rate of home wins and away "wins". Supposing teams i, j play one another with i at home, and $\alpha_i - \alpha_j = \Delta\alpha$ is small, then the basic BT model (without additional parameters) will predict

$$\Pr(i \text{ wins}) = \big(1 + \exp(\alpha_i - \alpha_j)\big)^{-1} \approx \frac{1}{2} - \frac{\Delta\alpha}{4}$$

And indeed if we choose to fit one with a constant home advantage factor, R estimates $h = -0.08$ with s.e. 0.11, which is very close to zero. An interaction between home advantage and team strength might make one useful however, and we see evidence of that in the exploratory analysis section.

Including Ties

As alluded to earlier, modifications can be made to the BT model such that ties can be included. A number of methods of accomplishing this have been suggested [27]. These more general models often use a different parametrisation. Instead of estimating α terms whose difference gives the odds, terms p_i are estimated such that $\sum_i p_i = 1$, and probabilities are calculated in different ways, shown below. One method for generalisation proceeds by estimating a tie-chance parameter p_0, as well as team-strength parameters $p_1, ..., p_n$. The chances of each outcome are then written as:

$$\Pr(i \text{ beats } j) = \frac{p_i}{p_i + p_j + p_0}$$

$$\Pr(i \text{ ties } j) = \frac{p_0}{p_i + p_j + p_0}$$

Another option (the Davidson model [28]) treats the probability of teams i, j tying as proportional to the geometric mean of the parameters estimated for each team:

$$\Pr(i \text{ beats } j) = \frac{p_i}{p_i + p_j + \nu\sqrt{p_i p_j}}$$

$$\Pr(i \text{ ties } j) = \frac{\nu\sqrt{p_i p_j}}{p_i + p_j + \nu\sqrt{p_i p_j}}$$

With ν representing a 'tendency towards ties'. The Davidson model can be extended to include a multiplicative home advantage parameter [29]. For team i playing at home, team j away:

$$\Pr(i \text{ beats } j) = \frac{\gamma p_i}{\gamma p_i + p_j + \nu\sqrt{p_i p_j}}$$

$$\Pr(i \text{ ties } j) = \frac{\nu\sqrt{p_i p_j}}{\gamma p_i + p_j + \nu\sqrt{p_i p_j}}$$

$$\Pr(j \text{ beats } i) = \frac{p_j}{\gamma p_i + p_j + \nu\sqrt{p_i p_j}}$$

A number of R packages exist for fitting models like these. `VGAM::brat` and related functions can fit ties-allowed models of the first kind. The problem with a ties-allowed model like this for our case is that the home advantage effect ought to then be accounted for, as including including all three match outcomes removes the home/away symmetry. After searching for a while we came across the `bpcs` package. While quite laborious to get working, this package allows one to fit ties-allowed models with a home advantage factor, along the lines of the third model type listed.

In the table below we list estimated strength parameters for each team with 95% highest posterior density bounds, as well as the estimates for the home advantage correction γ and the tie-factor ν. All values are given in log(...) form, as this transforms them onto the linear scale which is used for estimation.

Parameter	Mean	2.5%	97.5%
Wolves	−0.600	−2.070	0.840
Leeds	−1.830	−3.390	−0.320
South'ton	−2.320	−3.940	−0.780
Not'ham	−0.970	−2.490	0.470
Chelsea	−0.780	−2.260	0.690
Man City	2.860	1.220	4.520
Liverpool	0.990	−0.510	2.510
Crystal P.	−0.570	−2.040	0.920
Brighton	0.600	−0.900	2.100
Fulham	−0.390	−1.860	1.070
West Ham	−1.000	−2.490	0.490
Newcastle	1.620	0.100	3.140
Everton	−0.430	−2.000	0.970
Arsenal	2.090	0.450	3.630
Tottenham	0.770	−0.690	2.260
Brentford	0.580	−0.850	2.150
Aston V.	0.780	−0.740	2.270
Leicester	−1.440	−3.000	0.080
Man U.	1.440	−0.050	3.040
Bournemouth	−1.200	−2.720	0.290
gm	−3.760	−4.750	−2.800
nu	0.080	−0.130	0.290

As in the previous section the bounds on these values are quite wide, due to the number of matchups between any pair of teams being limited to 2. The constant effect $\ln(\gamma)$ is quite strongly negative, indicating an advantage towards the home team. Due to how we fit the model this reads more clearly as any "away-disadvantage" term, as will be shown momentarily. The tie-parameter $\ln(\nu)$ is very close to zero, and the 95% bounds on it do include zero. This indicates that tie chance $\propto \nu\sqrt{p_i p_j}$ is mostly determined by the respective team strengths. Taking a pair of teams as an example we can calculate win/draw/loss probabilities. As before using Leicester City playing away vs Newcastle United, we find team strength values of $\gamma p_{\text{LC}} = \exp(-3.76)\exp(-1.44) = 0.0054, p_{\text{NU}} = \exp(1.62) = 5.05$ respectively. Note that we multiply the away team win chance by γ. The tie term is $\exp(0.08)\sqrt{0.237 \times 5.05} = 1.18$, so the denominator in each case will be $0.0054 + 0.237 + 5.05 = 6.24$

$$\Pr(\text{Leic win}) = \frac{0.0054}{6.24} = 0.00087$$

$$\Pr(\text{Tie}) = \frac{1.18}{6.24} = 0.19$$

$$\Pr(\text{N.U. win}) = \frac{5.05}{6.24} = 0.81$$

Despite the possibly-unrealistic small chance of Leicester pulling through, this roughly matches the win chance for Newcastle we found in the previous section, 0.825.

Model Performance

In this section we will test the forecasting accuracy of these models. At the moment the models we've been looking at have been trained on the full 380 matches in the season, but for this section we will restrict this to the first 350 games, and test on the remaining 30. This is of course a fairly small sample size for testing, but given the bounds on the parameter estimates are already very wide, there's likely a steep tradeoff between more testing precision and model accuracy. We'll train both the regular BT model using `BTm` , as well as the constant-effect Davidson model using `bpc`. The testing dataset includes 30 games, of which 13 were home wins, 5 were ties and 12 were away wins. For the Bradley Terry model, we can tabulate the predicted vs actual results. Using a cutoff at $p = 0.5$:

		Actual	
		Home Win	Not
Pred	Home Win	3	4
	Not	10	13

We can see that the model is very keen to predict good results for the away team. Why exactly this happens we aren't too confident. The sample size is naturally very small, both for training and testing, so biased/innaccurate parameters are a possibility, as is a biased sample. Perhaps a different choice of cutoff could improve results, though as far as we're aware there's no theoretical justification for doing this. We can do the same for the three outcome model:

		Actual		
		Home Win	Tie	Away Win
Pred	Home Win	6	1	5

	Tie	7	4	7
	Away Win	0	0	0

As we can see, this model has the opposite behaviour. The low sample size makes it quite fraught to extrapolate from any results here, but the model seems fairly capable of predicting ties, assessing 4/5 correctly. Home wins are predicted either correctly or as ties, though the model has around even odds of choosing correctly. Away wins are where the model seems to struggle most. No match was predicted as an away win, and the model barely prefers predicting actual away wins as ties. In this case the culprit is likely the away-disadvantage constant γ. As we saw just above when fitting the first Davidson model, the magnitude of $\ln(\gamma)$ is far higher than any of the other constants, hence the γp_j terms in the numerator of away-team win chance are going to be overly small.

Between the negative results in the logistic regression section and now here, we seem to be struggling to find a good model for the win and loss patterns in football. As mentioned in the ordered logistic regression section, some degree of this is likely due to to poor information content in our data, but the failure to find a good predictive model in this section might suggest finding a new approach entirely.

Bradley-Terry based models with covariates included do exist and might be useful, depending on the choice of covariate. They would likely still run into the issue of sample size. If we wanted more useful results here, it might be wise to include data from previous leagues, perhaps weighted such that more recent matches have higher influence.

Appendix 7: Miscellaneous Scripts

Python script used for parsing text copied from the Mirror into valid R code defining the columns of a data frame.

```
# https://www.mirror.co.uk/sport/football/news/premier-league-derby-fixtures-dates-27249622
f = open("data.txt", "r")
lines = f.readlines()
rival_pairs = []
for line in lines:
    if line == "\n":
        continue
    else:
        parts = line.split(" - ")
        goodbit = parts[0]
        rival_pairs.append(goodbit)
replacements = {
    "Tottenham":"Tottenham Hotspur",
    "Man City": "Manchester City",
    "Man Utd": "Manchester United",
    "West Ham": "West Ham United",
    "Leicester": "Leicester City",
    "Brighton": "Brighton & Hove Albion"
}
fpairs = []
for i, pair in enumerate(rival_pairs):
    teams = pair.split(" vs ")
    if len(teams) == 1:
        print(f"error: string {teams} doesnt fit the pattern")
        continue
    team1 = teams[0]
    team2 = teams[1]
    if team1 in replacements.keys():
        team1 = replacements[team1]
    if team2 in replacements.keys():
        team2 = replacements[team2]
    newfpair = (team1, team2)
    if newfpair[::-1] in fpairs:
        continue
    else:
        fpairs.append( newfpair )
for i, (team1, team2) in enumerate(fpairs):
    print(f'{chr(i+65)} = c("{team1}", "{team2}"),')
```

Python script for converting R output from `summary()` into Typst tables.

```
#Example input, paste this into ./inp.txt
"""
Coefficients:
                                        Estimate Std. Error z value Pr(>|z|)
(Intercept)                             0.037884   0.195417   0.194   0.8463
home_or_awayHome                        0.106429   0.100773   1.056   0.2909
distance_groupingfarther               -0.229455   0.109013  -2.105   0.0353 *
points2021                              0.010394   0.002012   5.167 2.38e-07 ***
opponent_points2021                    -0.005595   0.002290  -2.443   0.0146 *
home_or_awayHome:distance_groupingfarther  0.322718   0.144729   2.230   0.0258 *
"""
#Then run python3 ./tablemaker.py
# Then check ./out.py and find:
"""
#table(
columns: 5,
table.cell([Coefficients:], colspan: 5)
[Variable], [Estimate], [Std. Error], [z value], [Pr(>|z|)],
[(Intercept)], [0.038], [0.195], [0.194], [0.846],
[home_or_awayHome], [0.106], [0.101], [1.056], [0.291],
[distance_groupingfarther], [-0.229], [0.109], [-2.105], [0.035],
[points2021], [0.010], [0.002], [5.167], [0.000],
[opponent_points2021], [-0.006], [0.002], [-2.443], [0.015],
[home_or_awayHome:distance_groupingfarther], [0.323], [0.145], [2.230], [0.026],
)
"""
def float_or_nop(s):
    # returns the input rounded to 3 d.p. if coercible to float, returns input unchanged if not
    # might be worth changing this to 3 s.f.
    try:
        out = float(s)
        return '{:.3f}'.format(out)
    except ValueError:
        return s
import os
os.system('dos2unix inp.txt') #rstudio console output uses DOS newlines, this converts to unix ones
os.system('dos2unix inp.txt') #only seems to be a problem when pasting using `p` in vim, you  can remove these
f = open("inp.txt",'r')

linesout = []
for line in f.readlines():
    if line == '':
        continue
    #split line into words
    parts = line.split(" ")
```

```
    # round floats, remove empty words, remove newlines
    parts = [float_or_nop(part.rstrip()) for part in parts if part != '']
    # 't value' should be kept in one piece, so we'll rejoin any z/p/t values we find
    # as well as any 'Std. Error's that we've split
    newparts = []
    i = 0
    while i < len(parts):
        part = parts[i]
        print(part)
        if part.lower() == "std." and i+1 < len(parts) and parts[i+1].lower() == "error":
            newparts.append(f"{part} {parts[i+1]}")
            i += 2  # Move to the next part after concatenating
        elif part.lower() == "error":
            i += 1  # Move to the next part without concatenating
        elif part.lower() in ["z", "p", "t"] and i+1 < len(parts) and parts[i+1].lower() == "value":
            newparts.append(f"{part} {parts[i+1]}")
            i += 2
        elif part.lower() == "value":
            i += 1
        else:
            if set(part) != {"*"} and part != "" and part != " ":
                newparts.append(part)
            i += 1  # Move to the next part
    newparts = [word.rstrip() for word in newparts if word !=" "]
    linesout.append(newparts)


#no. columns = max length of any line
col_count = max([len(line) for line in linesout])
with open("out.txt",'w') as fout:
    fout.write("#table(\n")
    fout.write(f"columns: {col_count},\n")
    for line in linesout:
        if len(line) == 1:
            # If line has one word it's probably a title
            # So we'll make it span all the columns
            if line[0] == "" or line[0] == " " or line[0] == "  ":
                continue
            fout.write(f"table.cell([{line[0]}], colspan: {col_count})\n")
            continue
        elif len(line) == col_count-1:
            # Often the variable has no title, so the column names have a length of one less
            # We'll add a "Variable" title, so all the row lengths match
            line = ["Variable"] + line
        fout.write("["+ "], [".join(line) + "],\n") #adds square braces for typst tables
    fout.write(")\n")
```

Almost automated backups of all project files to Google Drive.

```
#!/bin/zsh
# Little script for backing up project files, as well as updating the code in the appendices
# Used to have version control via Git written into this script but abandoned that - wasn't necessary


# ___ COPY CODE INTO WRITEUP ASSETS ___
# Code used for analysis copied into writeup folder, where it is rendered into the appendices
# Making copies means we can edit it a bit without compromising the actual files we use
echo "About to copy code into assets"
code_path="/Users/ralphbraithwaite/Documents/dissertation/code"
asset_path="/Users/ralphbraithwaite/Documents/dissertation/writeup/assets/code"
cp -fr $code_path $asset_path
echo "Code copied!"


# Main code files have newlines for readability
# but I don't want to have pages worth of the appendices taken up by blank spaces
echo "Removing blank lines in copied code"
cd "$asset_path"/code
for f in data_prep.R exploratory.R survival.R glm.R
do
  newfile="new"$f
  sed '/^\s*$/d' $f > $newfile #Removes blank lines
  sed '/^#/d' $newfile > $f #Removes commented lines - those that begin with '#'
  rm $newfile
  # No `sed -i` for inplace editing on Mac hence using $newfile as an intermediate
done
echo "Done!"
# ___ COPY ALL OBSIDIAN NOTES TO HERE ___
# Obsidian is a note taking app, notes are stored on Icloud but I want the relevant ones backed up with the rest of the project
obs_path="/Users/ralphbraithwaite/Library/Mobile Documents/iCloud~md~obsidian/Documents/Main/Uni"
here_path="/Users/ralphbraithwaite/Documents/dissertation/obsidian"

echo "About to copy files from Obsidian..."
cp -fr $obs_path $here_path
echo "Files copied!"
```

```
# ___ OPEN GOOGLE DRIVE ___
# Opens google drive for me, makes it very easy to upload my work
open -a /Applications/Arc.app https://drive.google.com/drive/my-drive

# I did have a git commit section in here but removed it since g-drive has version control
#
```

Word counter for source files, since it can be annoying to word count pdf files.

```
#/bin/zsh

# Since I'm not using Word it can be annoying to count the number of words in the project
# This file counts the number of words in specified source files and adds them up, then echos the section and total counts
# Since it's counting the source files it isn't perfect - syntax used for layout for instance will be counted
# Ideally we'd count the number of words in the pdf but this is enough

cd /Users/ralphbraithwaite/Documents/dissertation/writeup/sections
total=0
for file in litreview.typ exploratory.typ survival.typ regression.typ introduction.typ
do
  if [[ -e $file ]]
  then
    wcres=$(wc -w "$file")
    echo $wcres
    ((total+=$(echo $wcres | awk '{print $1;}')))
  fi
done
echo "The total is:  $total"
```

Appendix 8: Diary of work

```
# Week 1 -  Beginning Jan 15
At the beginning of this week I didn't have a topic - since I enrolled late for this term. I had a meeting on Friday with Housh where we discussed me starting a project on football modelling. Over the weekend I was moving into a new flat so I didn't have much time to work but I read the brief and looked around for data sources, and looked at a few of the papers he recommended.
# Week 2 - Jan 22
This week I've read a number of the papers recommended to me by Housh,
Began data collection using data from football-data.com, which includes by-match data for information such as participant teams, match result etc. I want data for the first goal times but can't find it on this website.
Data for distances between stadia isn't readily available so I compiled it myself using google maps and the R package geosphere.
I got data for previous season standings using  TNTSports.
I've begun putting the necessary data into a csv for use in analysis.
# Week 3 - Jan 29
Bit the bullet and paid for FootyStats, since they seem to be one of few data sources with information on in-match goal times that are accessible in csv form. fbref and whoscored.com both have the information but it's not downloadable and would require a lot of data entry.
Since the FootyStats data includes all the data I was getting from football-data I've replaced the latter. They agree on all the data they share but for instance team names are different between them, "Tottenham" vs. "Tottenham Hotspur" for instance. For this reason I'm just going to use the wider dataset for it all.
I've processed the data into a csv with the following columns:
kickoff timestamp - team name - game number - opponent - result - time to first goal - first goal censoring variable - distance travelled - home or away - red card given - derby indicator - last season league points - last season goal difference - last season wins
There are two rows for each match in the season, one where the  home team is listed under "team name" and distance travelled/ home or away will be 0/"Home" respectively, and another row where the away team will be found under "team name" and distance/ home away will be the actual distance/ "Away" respectively. This should be useful for certain analyses but it is very non-independent, so may be problematic for others.
Housh mentioned that derby games may be worth investigating - maybe teams play differently against their rivals - so I found a small list of premier league derbies online and included a boolean indicator variable on each match. Not sure if the list is comprehensive enough.
Started writing my literature review this week. Wrote a brief introduction and a summary of a paper on modelling home advantage by Pollard/Armatas. Read lots of papers and wrote brief summary notes on them. I've yet to decide on 3 other papers to focus on.

Asked Housh about the data duplication, what definition of derby game I should use.
# Week 4 - Feb 5
This week I have primarily focussed on the literature review. Lots of reading and selected the other 3 main papers to focus on - Prasetio/Harlili, Kharrat/ Boshnakov and Nero/Ritov. I've written 2000 words in total. After discussion with Housh he said it shouldn't be much more than 1500 so I will try to finish off the remaining parts - introduction and some detail on the K/B section, before pruning it a bit.

At this point my data on matches was primarily duplicated - each match in the season was represented on two rows, one home and one away. I made a new csv containing one row for each match so we are able to use either as necessary.
Housh sent me an email containing a list of derby games in the 22/23 season so I wrote a little python script to scrape these and put them in my R script as a factor variable.
I started a bit of exploratory analysis, with some crude tests for the effect of red cards on winning chance, and some boxplots for time-to-first-goal on a grouping of the teams from stronger to weaker. The boxplots didn't show much so I discussed it with Housh and he mentioned that boxplots aren't suited to survival data - better to use the survival graph for such statistics.
Next week I plan to start fitting linear models to the data, as well as tidying up my literature review and sending it to Housh for comments.
# Week 5 - Feb 12
After the friday meeting with Housh last week I sent off the first draft of the literature review and the first four entries of this diary to him.
While waiting for any feedback I've moved onto the exploratory analysis steps given in the project brief, developing plots to show differences in the results of various matches based on different factors, such as a binary grouping of teams based on last season's performance or home vs away performance, and looking at making clear and pretty survival plots for time to first goal, with features like medians and quartiles marked.
# Week 6 - Feb 19
I've been working more on survival analysis. Had some trouble getting `survplot()` to work for plotting hazard functions but eventually cracked it. Generated lots of plots of survival curves for different factors in the data.
`is_derby` doesn't seem to have much effect on first goal time - but the sample size is fairly small so there's little confidencein that.
I've had a rough cold this week so work has been slow and we skipped the meeting.

## Week 7 - Feb 26
Having made a number of plots and finding lots of useful descriptive statistics, I've begun writing up a section on the exploratory analysis. Motivation has come slow thanks to the lingering cold but it's a good start. I could do with finding a reference for how to structure it - a paper which lays out similar information maybe - since I struggle with getting the information into prose. Housh has sent back my first draft of the literature review with some notes for improvement, so maybe I'll implement those changes over the weekend or next week.

## Week 8 - Mar 4
I've taken most of Housh's suggestions on the literature review and made the changes. A few are impossible (one paper doesn't specify whether a scale parameter is 1, so I can't add it) but most were fine. I made a proper .bib file to manage the references, as I'd just been using plaintext notes before.
Made good progress this week on writing up some of the exploratory analysis and survival stuff. I've put all my work so far into a typst document so I can embed
```

images - it was getting annoying tabbing between the text and the images i was referring to.
I've done log rank tests on all the different survival curves - need to check over the assumptions for doing these and put some more analysis into them - test for proportional hazards etc.
Made some basic cumulative hazard plots, but I want to make these nicer, and test the curves they give for linearity.

Week 9 - Mar 11
Finished most of draft 1 for the survival analysis section. Finished up a lot of the analysis for the Weibull model section and a brief application of Cox regression. Wrote all of this up into my draft. Made a start on writing about logistic regression. Logreg seems much less interesting- fewer relevant variables among the ones I have.
Week 10 - Mar 18
Finished the cox regression section, it's quite short but wasn't meant to be an in depth analysis. Did more analysis for the logistic regression model and found the whole thing reduces to one variable, log odds $\propto$(point difference), and the predictions aren't even very good. I asked Housh whether to expand the survival analysis section instead or apply logistic regression to a different quantity, and he suggested using an ordered logit model, so I've made a start on that. Will hopefully finish in a day or two so I can submit this draft early.
(21st) - Finished sections on ordered logistic regression models and on bradley terry models. Gave the whole document some clean up and submitted draft 1 to Housh.
Week 11 - Mar 25
Housh has checked through some of the first draft and found an error i made in the logistic regression section so I'm redoing that. The resulting model seems to be much more interesting so I can probably write a lot more about it. Might move some of the Bradley terry models into appendices to make space.
Annoyed I made such a silly mistake but glad I got the first draft in early enough to give me time to fix it.

Week 12 - Apr 1
Finishing touches, adding small section on forecasting for logistic models and count models in the survival section. Going through and changing explanations so they make more sense, double checking some calculations etc.