STAT3612 Premier League Football Prediction

Grigoras Vlad, Parnes Jesse, Poon Sze Sen, Chan Tsz Hei

December 7, 2021

Introduction

When discussing a sports match there is always the urge to give some prediction of what you think will happen in the game. Sports betting taps into the satisfaction of predicting the outcome of matches and adds a monetary aspect which makes it all the more enticing for fans of the sport (Kampakis & Adamides, 2014). It is this art of prediction of sports matches (football, in our case) which we will try to investigate in this project. We chose to focus on football as it is the most popular sport in the world, and by far the most bet on sport in the world, making up 70% of the money spent on sports betting worldwide (Keogh & Rose, 2013). It is also the sport we are most familiar with so we can use some of our background knowledge to better interpret and understand our results.

Football betting has likely been around as long as the sport itself, which was created around the mid 1800s, although it's impossible to say for sure. Gambling worldwide has been legalised at many different times in different countries, and in many countries it remains illegal. It was illegal in the UK until the Betting and Gambling Act of 1960, after which many betting shops began to open. The football betting industry didn't boom until the 1990s, when Sky Sports began to broadcast multiple games each weekend and the Premier League was created. An increased exposure to football matches on TV nationally and worldwide, along with the advent of online bookmakers such as Bet365, led to more betting from the general public across the world. As time has gone by, more markets have opened for matches - where in the past one could bet simply which team would win and maybe what the score would be, one can now bet on statistics such as corners, yellow cards and possession, and make player-level predictions of number of shots on target and passes completed. The sports betting industry continues to grow, particularly in the U.S. where gambling is gradually being legalised state-by-state.

The use of machine learning to predict matches and create models to win money by gambling is increasing as well. For example, Twitter has been used to predict football matches, and bookmakers use their own machine learning models to set their odds.

Before we proceed, it is important we define some key terms, particularly gambling terms, that we will use and refer to throughout the report. In this report we use decimal odds. Decimal odds represent the amount you win (including your original stake) if you place a bet of one unit of currency. For example, if you bet $10 at odds of 3.5, you will receive 10 * 3.5 = $35 if you win the bet, which includes your original stake of $10, thus no decimal odds can be less than 1. For every odds there is an implied probability (or break-even probability) which is the percentage of the time you need your bet to win to break-even in the long run. To calculate implied probability for decimal odds you simply use $p = \frac{1}{odds}$, so a large odds implies a small probability of winning and vice-versa.

Our main objective for this project is to predict match results in the Premier League. Through research we quickly realised that beating the books, i.e. making predictions that would be accurate enough to make money through gambling, would be extremely difficult (Egglels et al., 2016). The odds are typically very accurate, particularly for premier league matches where so much money is bet on the games, and any poorly set odds would quickly have been bet into the right position by "sharp" bettors. On top of this, the vigorish means you have to predict significantly better than the books to win money. In light of this we decided to focus more on pure prediction of match results, without worrying about whether we would necessarily profit from gambling.

We were also very aware that to make money we would likely need a much larger dataset. For example, we were unable to consider rest time for teams (teams play in other competitions in midweek), individual player injuries/absences, managerial changes, poor weather etc., all of which we would expect to be important in match prediction. Our match prediction problem is one of multi-classification. There are three possible outcomes for any given Premier League match: a home win, an away win, and a draw.

Dataset

The Premier League football data was collected from Football-Data.co.uk by web scraping using Selelium. The collected data across Season 1993 to 2021 contains 10878 rows where each row represents a single football game. Each row contains the match statistics (e.g. teams played, goals scored, game result) and opening odds provided by betting companies *(See Table 1 for raw data frame (details in Appendix 1)).*

Date	Time	HomeTeam	AwayTeam	FTHG	FTAG	...	B365H	B365D	B365A	...
13/08/2021	20:00	Brentford	Arsenal	2	0	...	4	3.4	1.95	...
14/08/2021	12:30	Man United	Leeds	5	1	...	1.53	4.5	5.75	...
14/08/2021	15:00	Burnley	Brighton	1	2	...	3.1	3.1	2.45	...
...	...	...	...	...	...	...	...	...	...	...

Table 1: Raw Premier League football data frame.

The collected data was inspected to have certain inconsistency in terms of missing data and variables across the football Seasons. Hence, the data was cleaned before using it for visualizations, analysis, and model training.

Data Cleaning

From the raw data frame, 697 rows with no values were removed, and columns with no data were also removed *(See Appendix 4 for missing values profile).* The dataset was inspected to have inconsistent opening odds from different betting companies for each row. It was decided to stick with the opening odds provided by Bet365, Bet & Win, and Interwetten, and 3967 rows without opening odds provided by these companies were removed (all rows with Season before Season 2005/2006). The cleaned data frame $DF_{cleaned}$ contains 6214 rows and 33 columns *(See Appendix 2 for cleaned data frame).*

Data Preprocessing

Having aimed to create new predictors based on the features of $DF_{cleaned}$ for training (Season, Date, HomeTeam, AwayTeam, FTHG, FTAG, FTR), the first goal was to create new features *(See Table 2 for feature explanation)* named HTGS, HTGC, ATGS, ATGC, HTP, ATP, HS, AS, HST, AST, HF, AF, HC, AC, HY, AT, HR, and AR using *reference data frame* DF_{ref}. Denote T as the column indexes of DF_{ref} which are game days sorted ordered by week intervals $[t_0, t_1, .., t_{max-1}, t_{max}]$ where $t_i - t_{i-1} = 7$ days, $t_0 =$ the earliest game day in $DF_{cleaned}$, and $t_{max} =$ latest game day in $DF_{cleaned}$ plus minimum day $d \in \{ 1 \text{ to } 7 \}$

such that $(\sum_{i=0} t_i \; days) \bmod 7 \; days = 0$. Denote J as the row indexes of DF_{ref} are all the unique teams where every j_i represents a team in our dataset, and all the initial values for each row entry will be 0 *(See Table X for DF_{ref} illustration)*. Each new feature was generated with $DF_{cleaned}$ and a **fresh instance** (all values = 0) of DF_{ref}, and fill the values in DF_{ref} with one respective function *applyStat* to calculate the values for one new feature given a team $team_r$ and game day day_r of every row r in $DF_{cleaned}$ *(see Table 2 to see how respective applyStat is derived from the explanations of variables. In particular, we converted FTR to numerical scores before creating new features: for home teams (HTP), A = 0, H = 3, D = 1; for away teams (ATP), A = 3, H = 0, D = 1.)*:

$$DF_{ref}[\underbrace{\text{index of } team_r}_{j}, \underbrace{\text{index of } day_r}_{t}] \mathrel{+}= applyStat(DF_{cleaned}[\, team_r, day_r \,]) \text{ for r in rows.} \tag{1}$$

By taking the transpose of any instance of DF_{def} in (1), DF_{ref}^T contains the aggregated values of a feature with columns becoming the teams, and the indexes becoming the dates. DF_{def}^T now can be applied on a certain feature in DF_{ref} and used to generate a new feature from the mentioned using below algorithm:

```
vector newFeature;
for r in $DF_{cleaned}$:
   newFeature.push_back( $DF_{ref}^T[\underbrace{\text{index of } day_r}_{t}, \underbrace{\text{index of } team_r}_{j}]$ );
$DF_{cleaned}$\$newFeature = newFeature;
```

With new features (e.g. HTGS) in $DF_{cleaned}$, new features HTP, ATP, PtDiff were also computed (See Table 1 for formulas).

Feature	Explanation	Data type	Range
FTHG	Total goals scored by home team in that game	continuous	$[0, \infty]$
FTAG	Total goals scored by away team in that game	continuous	$[0, \infty]$
FTR	Game result	discrete	{H, A, D}
HTGS	Total goals scored by home team between t_i and t_{i-1}	continuous	$[0, \infty]$
HTGC	Total goals conceded by home team between t_i and t_{i-1}	continuous	$[0, \infty]$
ATGS	Total goals scored by away team between t_i and t_{i-1}	continuous	$[0, \infty]$
ATGC	Total goals conceded by away team between t_i and t_{i-1}	continuous	$[0, \infty]$
HTGD	HTGS - HTGC	continuous	$[-\infty, \infty]$
ATGD	ATGS - ATGC	continuous	$[-\infty, \infty]$
HTP	Scores obtained by home team in that game	discrete	{3, 1, 0}
ATP	Scores obtained by away team in that game	discrete	{3, 1, 0}
PtDiff	HTP - ATP	discrete	{-3, 1, 0, 1, 3}
HS (AS)	Total shots by home (away) team in that game	continuous	$[0, \infty]$
HST (AST)	Total shots on target by home (away) team in that game	continuous	$[0, \infty]$
HF (AF)	Total fouls by home (way) team in that game	continuous	$[0, \infty]$
HC (AC)	Total corners by home (away) team in that game	continuous	$[0, \infty]$
HY (AY)	Total yellow cards by home (away) team in that game	continuous	$[0, \infty]$
HR (AR)	Total red cards by home (away) team in that game	continuous	$[0, \infty]$

Table 2: Variable Explanations (∞ is used for simplicity to describe the bounds as actual goals scored (conceded) by each team in games could vary. In reality, the numbers obviously will be not be that large.)

To prevent look forward bias in model training, the current features in $DF_{cleaned}$ cannot be used as predictors (i.e. we cannot predict the final scores at the first minute of game) (Tax & Joustra, 2015).

	2005-08-13	2005-08-20	2005-08-27	...	2021-11-20	2021-11-27	2021-12-04
Arsenal	0	0	0	0	0	0	0
Aston Villa	0	0	0	0	0	0	0
Birmingham	0	0	0	0	0	0	0
Blackburn	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...

Table 3: Reference Data Frame DF_{ref}.

Thus, cumulative values for newly created features (e.g. HTGS, HTGC) are created as predictors which are reset to 0 at the start of each Season. A cumulative value v_i of a feature at t_i is calculated by $v_i = \sum_{t=0}^{i-1} valueOfFeature_t$, which means that $v_i = 0$ at t_0. In this way, when we are passing the values for predictions at t_i, the model can never know the $feature_i$ value (data in the past), and the look forward bias is eliminated. To get the average performance of a team in a given Season, we scaled the cumulative values to standardize every cumulative value v to $v_{standardized} \in [0, 1]$ such that

$$v_{standardized} = v/ \text{ number of games played by a given team in a given Season} \tag{2}$$

Besides the standardized cumulative values created, we came up with HStreak, AStreak, HPrevPos, and APrevPos. HStreak (AStreak) are the trend measures of home teams' (away teams') match results at *t-1* time *(See Appendix X for algorithm used to generate HStreak (AStreak))*. For example, if a home team has a HStreak of 5 (-5) at t_i, it means that it has won (lost) 5 games in a row before t_i (could be came across more than 1 Season). HPrevPos (APrevPos) are the ranking measures of home teams (away teams) *(See Appendix X for algorithm to generate HPrevPos (APrevPos))*. Note that the highest value for HPrevPos (APrevPos) is 41 which is just $|J|$ (number of unique teams), and the lowest will be 0 (only happens if a team at t_i was not in Premier League in the previous Season of t_i). Assuming there are only 20 teams, among 41 teams in $DF_{cleaned}$, 21 teams will have HPrevPos (APrevPos) of 0 in a Season. For example, if an away team has a APrevPos of 41 at t_i, it means that the team was the first-ranked team in the previous Season of t_i. In contrast, if a home team has a HPrevPos of 0, it means that it was not in the Premier League in its previous Season.

Last but not least, the odds provided by three companies were replaced by the mean odds of the three instead as three sets of odds they share a very similarity in their given odds. We then combined the odds features with $DF_{cleaned}$ into $DF_{preproceesed}$ *(See Appendix 3 for $DF_{processed}$ details)*. We split $DF_{processed}$ into training set (80%, n = 4971) and the testing set (20%, n = 1243). While the former set was used for Exploratory Data Analysis (EDA), model training, and model evaluations, and the latter was used for model performance analysis and evaluations. For this work, FTR was used as response variable, and all the new features we have created being the predictors.

Exploratory Data Analysis

EDA is conducted to explore the distribution of data and the correlation between variables in the training set *(See Table 4 for raw counts for the training set)*.

The correlation value of 1 or -1 indicates two variables have strong positive or negative correlations, while 0 indicates the variables are independent to each other. If the correlation is too high, it means the linearity assumption is violated which can lead to multicollinearity results, which can be very difficult to interpret. From Figure 1, we can see most correlation values lies around 0.5 or lower, which appear to be normal.

Name	Counts
Rows	4971
Continue Columns	28
Discrete Columns	4

Table 4: Raw counts for the training set

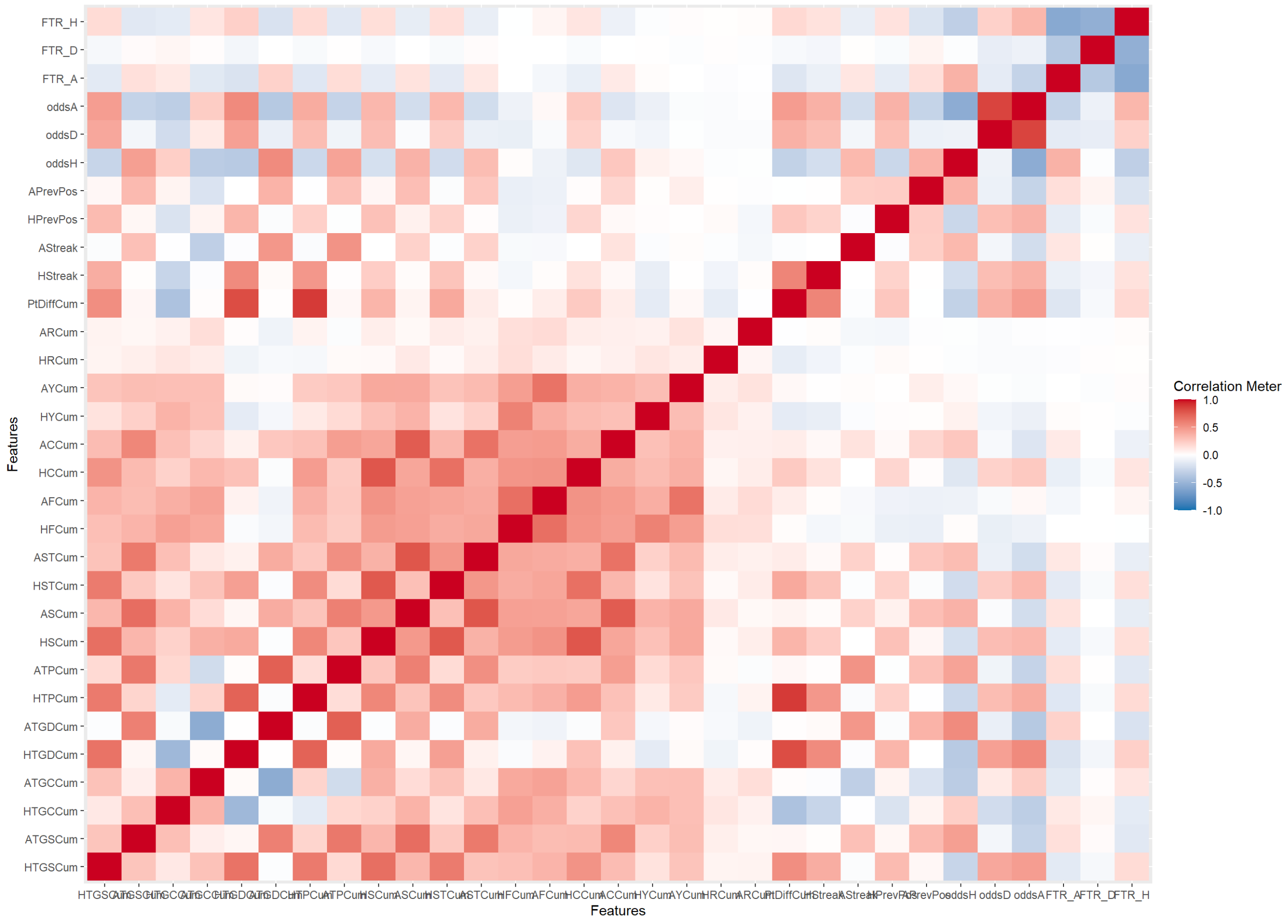
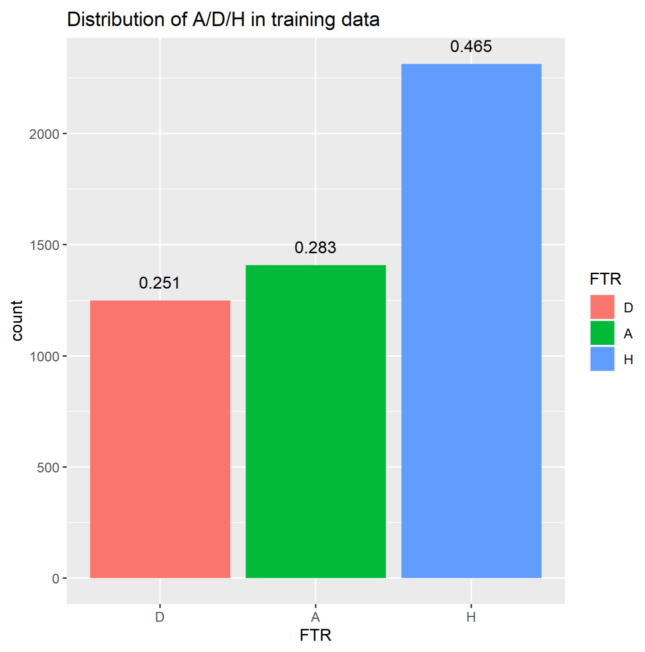


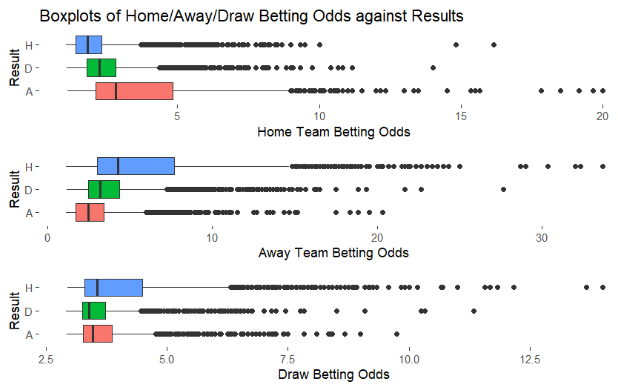
Figure 1: Correlation Matrix of the predictors in training set

Figure 2 shows the distribution of our response variable FTR. From the figure, we see that teams playing at Home have highest relative probability of winning (0.465), followed away team wins, and draw. It means that, if we solely bet on home teams for every game, we have an estimated chance of 46.5% to win the bet. Thus, we treat 46.5% as our baseline accuracy, and we aimed to train the models with an accuracy higher than 46.5%.

Using the box plots in figure 2(b), we can observe that there is an interesting relationship between the results of the game and the odds of the away/home team winning. Essentially, when the game is won by the home team, the home team also tends to have smaller betting odds than the away team (meaning that the betting company deemed it more likely to win). This relationship can also be observed with regards to the away team victory, which seems to be correlated with higher odds for the home team. Interestingly, when we plot the odds of draw against the full time results, the relationship is not as significant, but there seems to be a slightly higher mean odds for the home team.



(a) Distribution of Response variable FTR



(b) Distribution of Response variable FTR against odds

Figure 2: Distribution of Response variable FTR

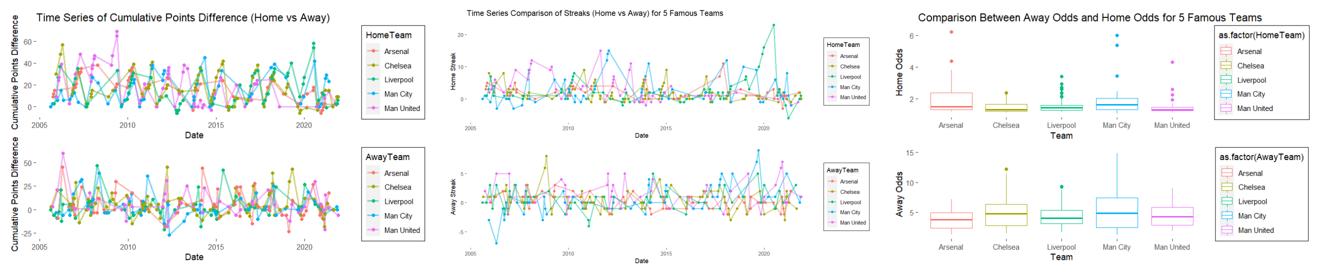
We have decided to select five teams that have consistently performed exceptionally well and explore a few time series plots in order to get accustomed to the variables and the potential relationships between them. First we can observe the Cumulative Point Difference tends to heavily fluctuate, but seems to sit above 0 for well-performing teams. There also seems to be a difference when the team is playing away compared to when it is playing home.

A very interesting observation that can be drawn from figure 3(b) is that even historically well-performing teams tend to have a poorer performance when playing in a stadium that is not their home stadium. The away streak seems to more often be around 0-5, rather than the 5-10 (and even up to 20) for the home streak. Given that betting odds seem to be lower the more likely a team is to win, there might be a relationship between the status of the team (away/home) and the betting odds.

Building upon our previous findings, we have plotted box plots of these 5 famous teams and their betting odds when playing both home and away. Comparing the two plots, we can clearly see a relationship between the familiarity of the stadium that the team is playing in and the betting odds set by companies. This effect seems to be mediated by the performance of the team, which is also affected by the stadium that the team is playing in (as shown in figure 3(a)).

Model Training

We used multinomial logistic regression (Multi LogitR), Support Vector Machine (SVM; radial kernel), Extreme Gradient Boosting (XGBoost) Linear, and XGBoost Tree algorithms. R package caret was used to implement those algorithms to train the models with 5-fold cross validation (with no repetition). For *(See Appendix X for model training codes).*



(a) Cumulative Points Difference (b) Streaks (c) Odds

Figure 3: (3(a) & 3(b)) Time Series plots of variables and (3(c)) Odds comparisons of 5 popular teams in Premier League.

Results

Model analysis

Table 5 shows the accuracy of each performing calculated using below formula. The results show that all models have better performance than our baseline accuracy (solely guessing the home team as the winner has 46.5% chance to win the bet). In particular, XGB Linear has the highest accuracy of 92.92% among other algorithms.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Model	Accuracy
Multi LogitR	53.85%
SVM	55.52%
XGB Linear	92.92%
XGB Tree	55.88%

Table 5: Model accuracy

Confusion Matrix (From tables 6 to 9) with columns denote the ground truth of and rows denote the predicted response variables $F\hat{T}R$. We can see Multi LogitR and SVM cannot predict or misclassified FTR with draw outcomes, while XGBoost algorithms could better generalize all three possible outcomes.

Model Statistics of specificity, sensitivity,and precision were computed for 4 algorithms as shown in tables 10 to 13 (). From the sensitivity ($\frac{TP}{TP+FP}$) results, we double confirmed that the Multi LogitR and SVM algorithms are not sensitive to the draw games at all with respective sensitivity of 0 and 0.0064, while they are better in predicting the home games with higher sensitivity. We can also calculate FP from $1 - Specificity$, we also confirmed that XGB tree algorithm is performing well in avoiding false positives.

Hyper-parameters tuning analysis was conducted investigate to each model's parameter can be tuned to obtain highest accuracy. For Multi LogitR model (figure 4(a)), we discovered that choosing a regularization parameter λ of 0.0342 could obtain a better accuracy compared other 2 parameters in and green. From figure 4(b), the cost parameter C for SVM optimizes the accuracy when it is set to 1. For XGB Linear model, model accuracy could vary by tuning alpha (L1 regularization), L2 regularizing parameter, and boosting iterations. For example, choosing boosting iterations of 140 and L2 of 0.0001 could result in the best accuracy among the other choices L2 regularization parameters. For XGB Tree model, parameters

	D	A	H
D	0	0	0
A	213	581	218
H	1019	827	2096

Table 6: Multi LogitR

	D	A	H
D	8	1	1
A	284	679	240
H	957	728	2073

Table 7: SVM (Radial)

	D	A	H
D	1052	16	6
A	78	1315	56
H	119	77	2252

Table 8: XGB (Linear)

	D	A	H
D	120	87	73
A	306	701	284
H	823	620	1957

Table 9: XGB (Tree)

	D	A	H
Se	0.00	0.41	0.91
Sp	1.00	0.87	0.31
Pr	N/A	0.56	0.53

Table 10: Multi LogitR

	D	A	H
Se	0.01	0.48	0.90
Sp	1.00	0.85	0.37
Pr	0.80	0.56	0.55

Table 11: SVM (Radial)

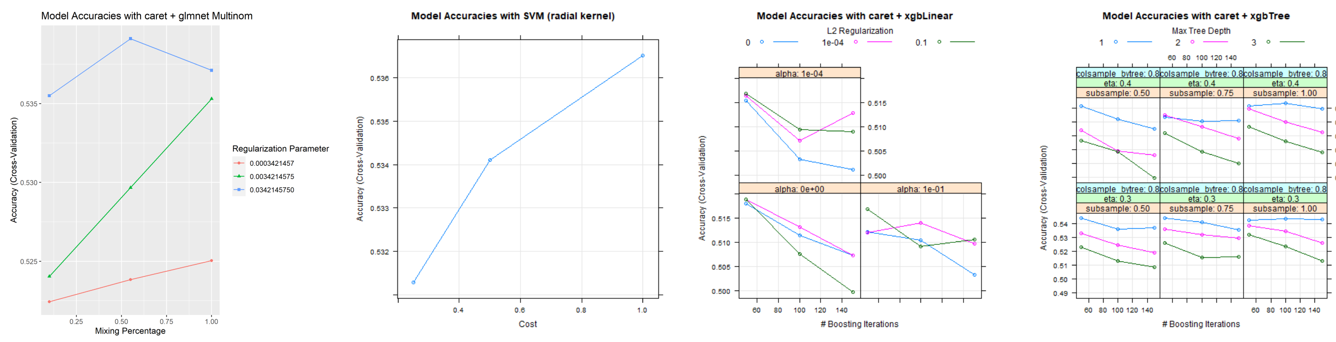
	D	A	H
Se	0.84	0.93	0.97
Sp	0.99	0.96	0.92
Pr	0.98	0.91	0.92

Table 12: XGB (Linear)

	D	A	H
Se	0.10	0.50	0.81
Sp	0.96	0.83	0.46
Pr	0.43	0.54	0.58

Table 13: XGB (Tree)

like *eta* and *Max Tree Depth* could be tuned. It is observed that choosing a *Max Tree Depth* of 1, the highest model accuracy could be obtained by setting *eta* either as 0.3 or 0.4.



(a) Multi LogitR (b) SVM (c) XGB Linear (d) XGBoost Tree

Figure 4: Model Accuracy against Hyper-parameters tuning for each algorithm

Model Performance

We measured the model performance by computing mean AUC (Area Under ROC Curves) and the log-loss *(See table 14-15 for details)*. In terms of AUC and Log loss, Multi LogitR and XGB Tree the best among other algorithms which have a mean AUC of 0.68 and mean log loss of 0.96.

Discussion

All models are found to have a better performance than solely guessing the home team as the winner of matches. It is even more surprising to see our XGB Linear model having a accuracy of 93%. However, it was found that the XGB Linear fit had a problem of overfitting into traning set, which was evidenced by its performance in terms of relative lower AUC and higher log loss compared to other models. The model could be improved by further tuning the hyper-parameters to model accuracy and performance.

The notion of using cumulative values as predictors instead of using single predictor is important. Given single match statistic being used as predictors The of accuracy SVM on the training is 0.97. The reason of being this impossibly accurate is that the training based on future data could cause look-forward bias. In other words, it's impossible to obtain such a high accuracy in our real world. Instead, having used the

	min	median	mean	max
Multi LogitR	0.66	0.68	0.68	0.72
SVM	0.63	0.63	0.64	0.64
XGB Linear	0.65	0.66	0.66	0.69
XGB Tree	0.67	0.68	0.68	0.71

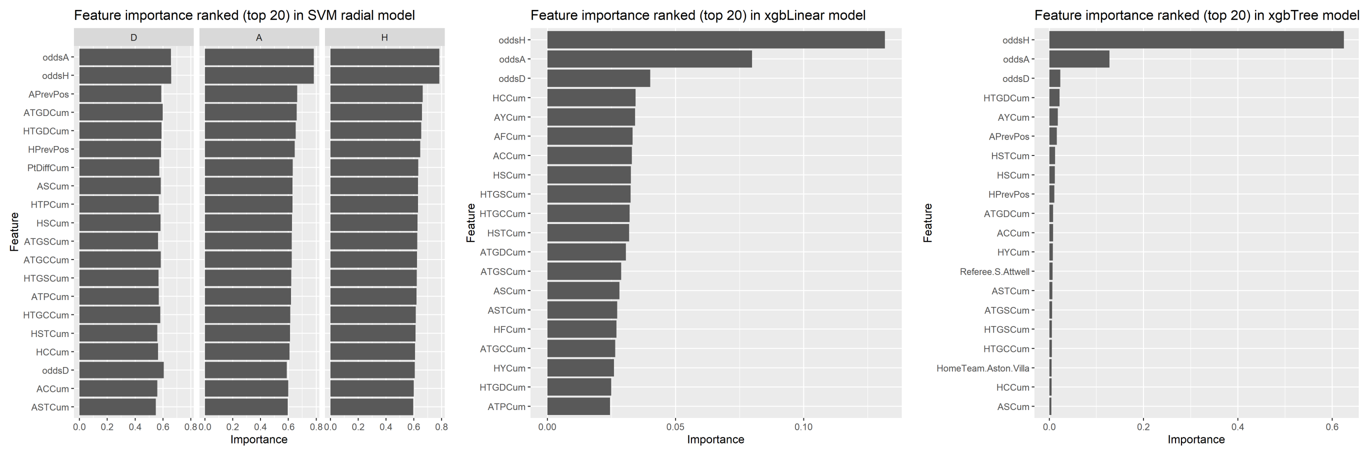
Table 14: AUC Metric on resampled training set

	min	median	mean	max
Multi LogitR	0.58	0.97	0.96	0.98
SVM	0.58	0.99	0.99	1.00
XGB Linear	0.59	1.05	1.04	1.05
XGB Tree	0.59	0.97	0.96	0.98

Table 15: Log loss Metric on resampled training set

cumulative values as predictors, we are not able to look into future, and therefore the accuracy decreased to around 55% on training set. The significance drop in accuracy between two versions reflects that there still exists several kinds of uncertainties in actual football betting.

Moreover, from figure (5), we discovered that the odds feature are most important features across different models. It means that the odds have significance for models to predict the match result. However, we did not do much about the feature creations using the odds data, like creating the cumulative values of Odds, and therefore we believe that the model accuracy could have a room for improvements if we have done data transformation on the odds data.



(a) SVM (b) XGB Linear (c) XGBoost Tree

Figure 5: Feature importance analysis of model algorithms

Limitation

Firstly we come up with the insufficient knowledge in understanding all hyperparameters of the implemented algorithms, especially for more complicated models like XGBoost. Therefore the models we have developed might not be optimized and reflecting the true capabilities of the algorithms. Also, plotting ROC curves in multi-classification task poses a very challenging time to us, and we could not figure out the appropriate interpretations and plots for ROC curves even we have computed AUC based on programming packages. Furthermore, when we were training models repeated 5-fold cross-validation as we found it took so long for our computing resources to complete the training, and therefore we used non-repeated 5-fold cross-validation. Last but not least, we have tried to use ensemble method for stacked-model training, however we could not figure out the correct formats of inputs and overall parameters. Therefore, we could not evaluate the model performance of the ensemble models.

Conclusion

Our model met our hypothesis that using ML models to predict match results is better than solely betting on home teams, and this is approximately what we expected heading into the process. Just blindly predicting a home win will yield you the correct result 46.5% of the time, and our models are better than this baseline measure. As mentioned in the introduction, there were many key data points that would have been useful to include in our predictions, both player-level and team-level, which are missed in the data we collected. Given there is a number of football prediction models recently have started to use advanced statistics to enhance their predictions, it is likely to see if our models would have better performance with these data (Baio & Blangiardo). For example, the most popular advanced statistic at the moment is xG (Expected goals), which estimates how likely a shot is to end up being a goal based on the position on the pitch from where the shot is taken, the body part used to strike the ball and more (Brechot & Fleepp, 2020). These are all good for understanding how well a team truly played during a match, and are used in models all the time, for example in FiveThirtyEight's model for football prediction which uses adjusted goals (which places less emphasis on goals scored in favourable conditions, such as a goal scored when a team has more players on the field), shot-based expected goals and non-shot-based expected goals (FiveThirtyEight, 2020).

Ultimately football is a game with typically very few goals, which means luck plays an extremely important factor. No matter how good a model is, it will still be wrong a lot of the time, and this is part of the reason why the sport is so captivating. If there was no chance of an upset, no one would want to watch.

References

Baio, G., & Blangiardo, M. (2010). Bayesian hierarchical model for the prediction of football results. *Journal of Applied Statistics*, 37(2), 253-264.

Brechot, M., & Flepp, R. (2020). Dealing with randomness in match outcomes: how to rethink performance evaluation in european club football using expected goals. *Journal of Sports Economics*, 21(4), 335-362.

Eggels, H., van Elk, R., & Pechenizkiy, M. (2016). Expected goals in soccer: Explaining match results using predictive analytics. *In The machine learning and data mining for sports analytics workshop* Vol. 16.

FiveThirtyEight. (2020).*How Our Club Soccer Predictions Work*. Retrieved from https://fivethirtyeight.com/methodology/how-our-club-soccer-predictions-work/.

Kampakis, S., & Adamides, A. (2014). *Using Twitter to predict football outcomes.* arXiv preprint arXiv:1411.1243.

Koegh, F. & Rose, G. (2013). *Football betting - the global gambling industry worth billions.* BBC Sport.

Spoonnvidia. (2021). *GitHub Repo: spoonnvidia/stat3612*. Retrieved from https://github.com/spoonnvidia/stat3612

Tax, N., & Joustra, Y. (2015). Predicting the Dutch football competition using public data: A machine learning approach. *Transactions on knowledge and data engineering*, 10(10), 1-13.