Forecasting Premier League Football Outcomes Using Machine Learning

Nikolai Tennant-Stambler

Department of Data Science Brown University

DATA 1030: Hands on Data Science

Professor Zsom

December 9th, 2022

GitHub Repository

Introduction

My goal was to forecast Premier League football (hereafter PL) match outcomes. I used PL season data for 2021/2022 and 2022/2023. Football is the most funded sport in the world, with this year's World Cup costing $220 billion.[1] Bettors stake an average of £2.7 million per PL match, and the Premier League's clubs have billions of followers.[2] Reliably forecasting winners could be helpful in betting, fantasy leagues, and overall game enjoyment.

My data consists of eight features and 457 data points (matches) before lagging and preprocessing. The features are Referee, B365H, B365D, B365A, HICT, AICT, DifICT, and Result (target). Referee (hereafter R) is a categorical variable that stores each match's referee. I downloaded this data from an intermediary holding PL Season record.[3]

B365H, B365D, and B365A are continuous variables from 365Bets included in the PL match record intermediary data.[4] Each shows the likelihood that its corresponding postfix will occur: (H = Home-Win, D = Draw, A = Away-Win). Their values govern the amount received per each $1 wagered.

The continuous features HICT (Home) and AICT (Away) reflect the strength of each team, while the DifICT shows the difference in strength between the teams. I derived scores from the Fantasy Premier League (FPL) ICT index, which is available in a GitHub repository.[5] ICT scores reflect a player's performance. Initial team ratings come from a Kaggle with FIFA 22 or 23 data. Summing ICT scores after each match updates team ratings.[6] A positive DifICT indicates a stronger Home-side and vice-versa. Result is a multi-classification target variable with three outcomes: Home-Win/Away-Loss (2.0), Draw (1.0), and Home-Loss/Away-win (0.0). This data comes from PL match records.[7]

University of Mumbai data scientists used the same dataset for PL matches.[8] They achieved 67% accuracy with an SVM model.[9] Medium used the same data to develop a GRU model with 92% accuracy.[9] The author found that sequential models perform better for football predictions.[10] The Mumbai scientists found more recent data is more beneficial and positively affects model accuracy.[11]

[1] "The Cost of Hosting a FIFA World Cup," Deccan Herald, September 25, 2022

[2] IResearch, ASSESSMENT OF THE RELATIONSHIP BETWEEN SPORTS BETTING AND VIEWERSHIP OF ENGLISH PREMIERSHIP LEAGUE (EPL); Harries, "Premier League Clubs RANKED by Popularity across social media

[3] England Football Results Betting Odds | Premiership Results & Betting Odds

[4] Premiership Results & Betting Odds

[5] Vaastav/Fantasy-Premier-League: Creates a .csv File of All Players in the English Player League with Their Respective Team and Total Fantasy Points

[6] Fantasy Premier League, Official Fantasy Football Game of the Premier League

[7] Premiership Results & Betting Odds

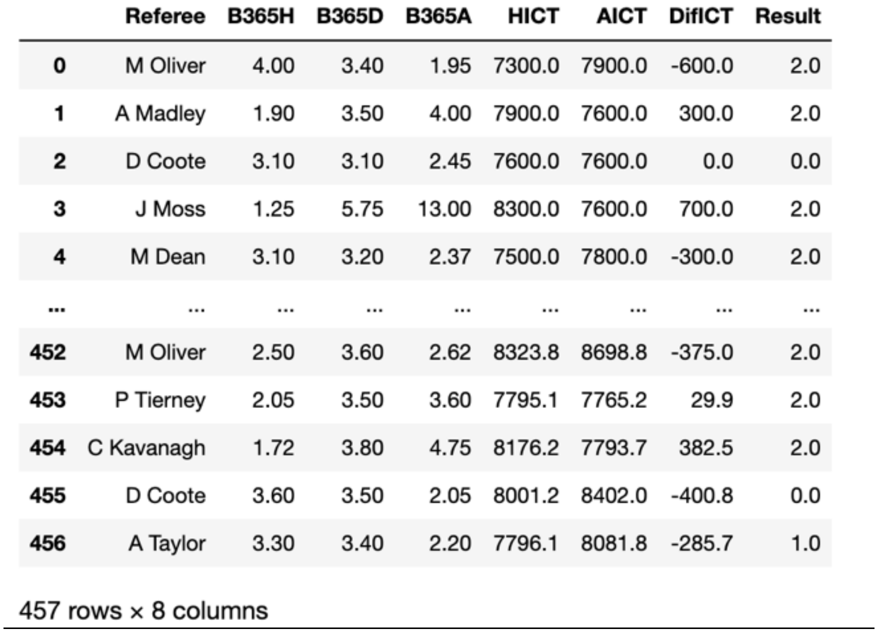
[8] Ajgaonkar, Yash, Anagha Patil, Kunal Bhoyar, and Jenil Shah, "Prediction of Winning Team Using Machine Learning"

[9] "Prediction of Winning Team Using Machine Learning"

[10] "Prediction of Winning Team Using Machine Learning"

[11] "Prediction of Winning Team Using Machine Learning"

Exploratory Data Analysis



	Referee	B365H	B365D	B365A	HICT	AICT	DifICT	Result
0	M Oliver	4.00	3.40	1.95	7300.0	7900.0	-600.0	2.0
1	A Madley	1.90	3.50	4.00	7900.0	7600.0	300.0	2.0
2	D Coote	3.10	3.10	2.45	7600.0	7600.0	0.0	0.0
3	J Moss	1.25	5.75	13.00	8300.0	7600.0	700.0	2.0
4	M Dean	3.10	3.20	2.37	7500.0	7800.0	-300.0	2.0
...	...	...	...	...	...	...	...	...
452	M Oliver	2.50	3.60	2.62	8323.8	8698.8	-375.0	2.0
453	P Tierney	2.05	3.50	3.60	7795.1	7765.2	29.9	2.0
454	C Kavanagh	1.72	3.80	4.75	8176.2	7793.7	382.5	2.0
455	D Coote	3.60	3.50	2.05	8001.2	8402.0	-400.8	0.0
456	A Taylor	3.30	3.40	2.20	7796.1	8081.8	-285.7	1.0

457 rows × 8 columns

Figure 1: Shows pre-preprocessed and lagged data in a 457x8 matrix. Each row is a match.

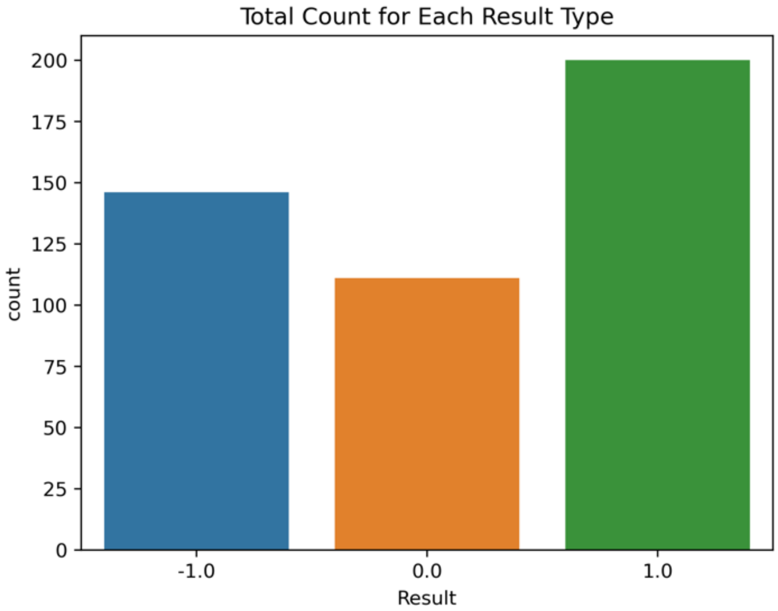


Figure 2: Shows occurrences in each result type. These include Home-Win – 200 (43.8%), Away-Win – 146 (31.9%), and Draw – 111 (24.3%). Fig. 2 illustrates a clear statistical advantage for home, thus proving the fabled home-field advantage.

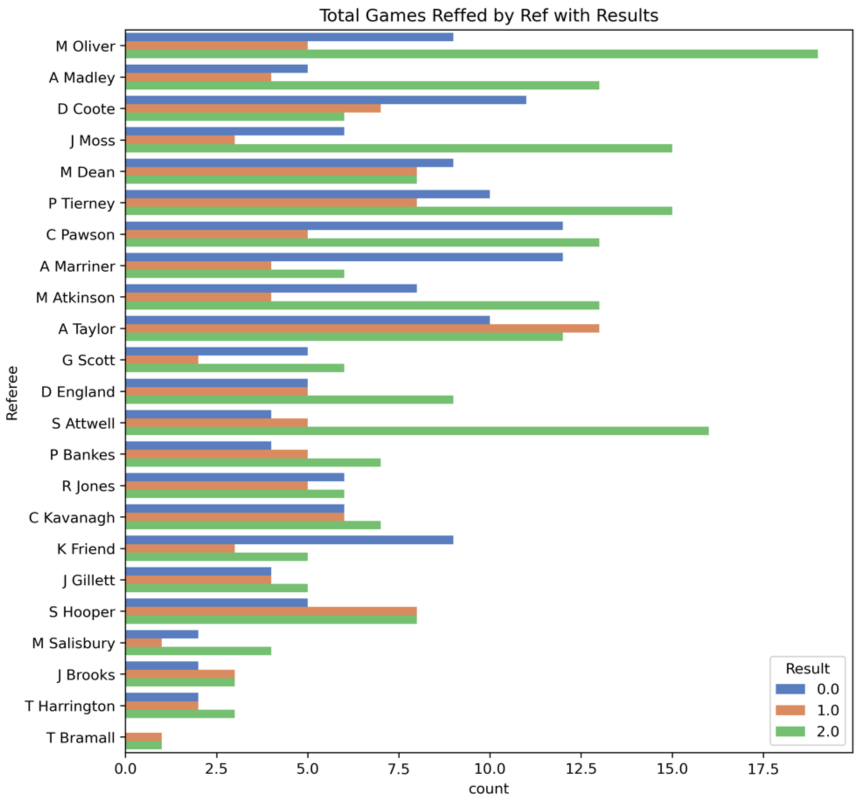


Figure 3: Shows # of games for each ref and their results. This graph visualizes the R feature variable and gives insight into the likelihood of achieving each result type (A, D, H) with each referee.

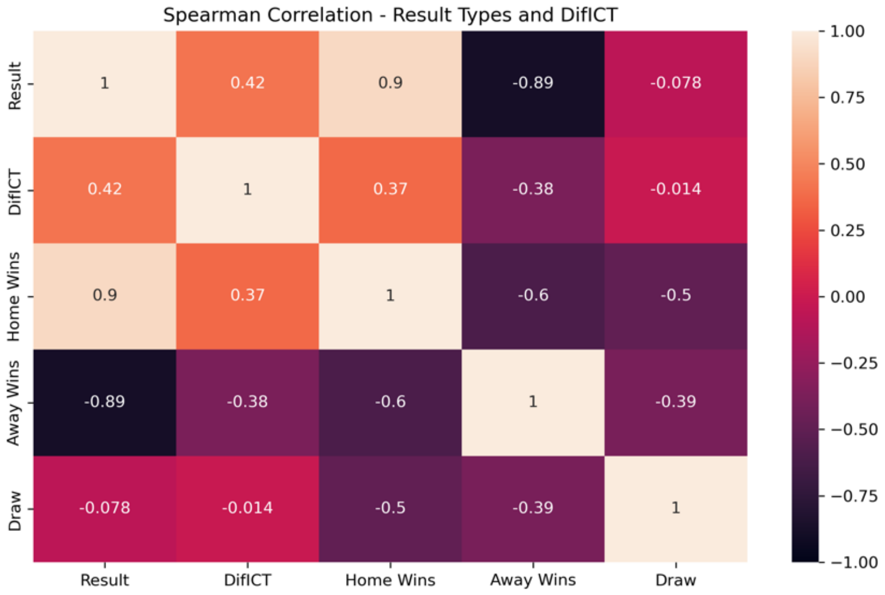


Figure 4: Shows the spearman correlation between DifICT and different outcomes. The correlation values vary between -0.014 and 0.9. The most important are DifICT to result (0.42), Home win (0.37), Home Loss (-0.38), and Draw (-0.078). There is a medium positive correlation between DifICT and results, showing the stronger the team, the more likely they will win. There is no correlation to Draw.

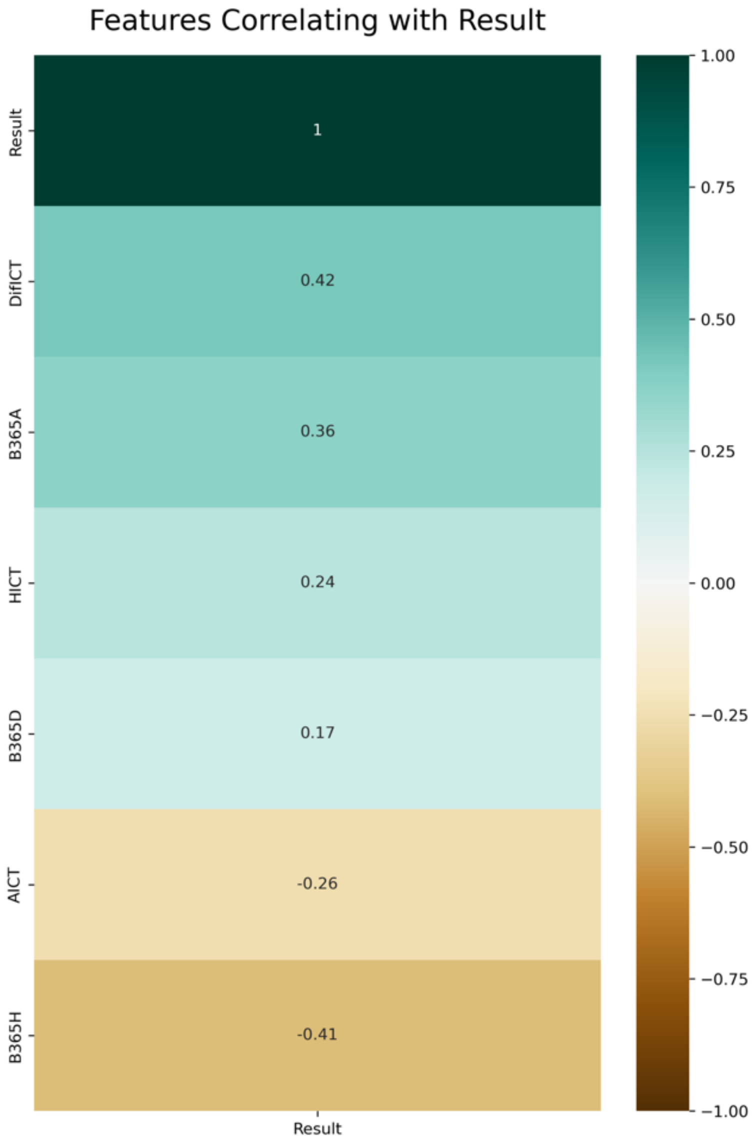


Figure 5: Shows the correlation between all features (except R) and Result.

Methods

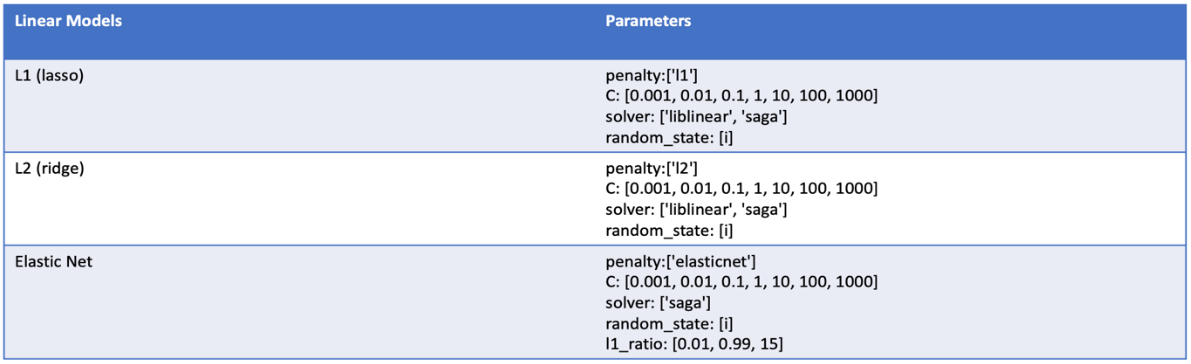
The final dataset is non-IID time series data without group structure. Therefore, features must be accessible at the start of each match; otherwise, predictions are based on future data. The data passes this test. R is posted days before the match, the AICT/HICT/DifICT are updated with the values from the previous match, and betting odds are announced by the day of the match. Time series data must Lag values to make the values appear contemporaneous. I did four different Lags: 1, 3, 5, and 7 to show the effect of differing lags on performance. I cut the first *n* number of rows to account for the newly created NaN values, creating four matrices sized: 450x13, 452x25, 454x37, 465x49.

Next, I created four different preprocessors, one for each lag. I used a StandardScaler for continuous features, except HICT and AICT, as they had normal or tailed distributions and were not bounded. I used OneHotEncoder for the R as this was unordered categorical data. I used MinMaxEncoder for HICT/AICT because these continuous features are bounded (73000-86309) - these numbers only seem unbounded because they are scaled up 1000x.

I used seven different models to train the data, including: SVC, Logistic Regression (with separate Lasso, Ridge, and ElasticNet penalties), CATBOOST, LightGBM, and Random Forest. Each model had a separate ML pipeline but were identical except in the parameters they accepted because TimeSeriesSplit lacks a random state parameter since order matters when splitting time series data. Thus, one cannot shuffle the data randomly. Instead, TimeSeriesSplit splits the data into multiple training and test sets. I used two splits. The parameters are shown in the tables below:

Booster Models	Parameters
CATBOOST	depth:[3, 1, 10] random_seed: [i] iterations:[1, 10, 25] learning_rate:[.01, 0.03] l2_leaf_reg:[5, 10, 100] border_count:[32, 5] thread_count:[4]
LightGBM	n_estimators: [5] num_leaves: [3,6,8,31] max_depth: [40] colsample_bytree: [0.99] min_child_samples: [5,20,25,50] min_child_weight: [1e-5, 1e-3, 1e-2, 1e-1, 1, 1e1, 1e2, 1e3, 1e4] random_state: [i]

Other Models	Parameters
SVC	C: [1, 3, 10, 30, 100] kernel: ['linear'] gamma: ['auto', 'scale'] degree: range(1, 6, 1) probability: [True] random_state: [i]
Random Forest	max_depth: [10] min_samples_leaf: [1,3,5] n_estimators: [50,100,200,500] max_features: ['auto', 'sqrt', 'log2'] random_state: [i]



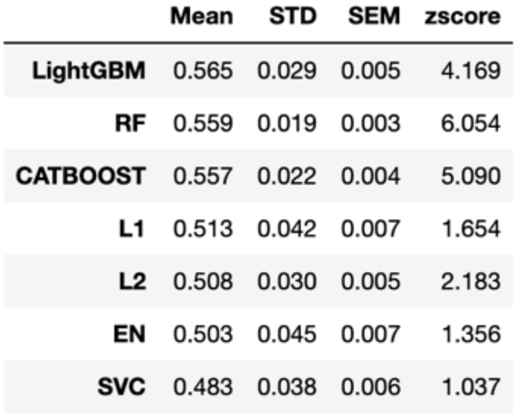
Linear Models	Parameters
L1 (lasso)	penalty:['l1'] C: [0.001, 0.01, 0.1, 1, 10, 100, 1000] solver: ['liblinear', 'saga'] random_state: [i]
L2 (ridge)	penalty:['l2'] C: [0.001, 0.01, 0.1, 1, 10, 100, 1000] solver: ['liblinear', 'saga'] random_state: [i]
Elastic Net	penalty:['elasticnet'] C: [0.001, 0.01, 0.1, 1, 10, 100, 1000] solver: ['saga'] random_state: [i] l1_ratio: [0.01, 0.99, 15]

Figures 6: Tables with parameters used for each model

The models used internal random state parameters to account for the lack of random state in TimeSeriesSplit. I placed all parameters inside a for loop, which looped through different random states. These random states were: 1, 10, 42, 60, and 90.

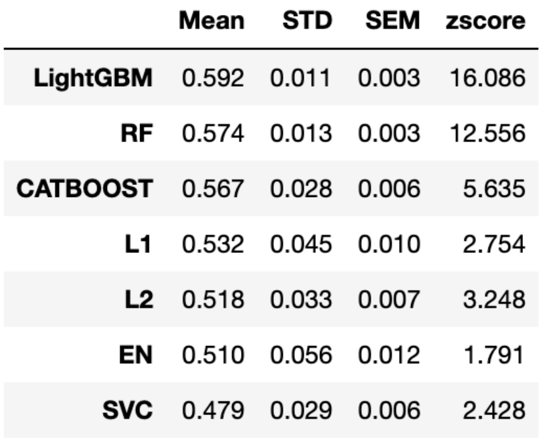
I passed the parameters into the parameter_gird of GridSearchCV, which hyper-tuned all parameters automatically. “F1_micro” was the scorer for the validation set, and every model output was saved in a list of models. Two-hundred-eighty models were saved through the different lags, splits, and random state combinations. The evaluation metric for these models was accuracy. I chose this because the dataset is balanced, and the goal is for the model to predict the correct result accurately; thus, knowing how well the model achieves this goal is important. I calculated all test scores’ standard deviation and means to account for uncertainties, finding LighGBM, RF, and CATBOOST the best models as they scored highest in mean and lowest in STD, but deciding LighGBM was the best model.

The following tables show the results:



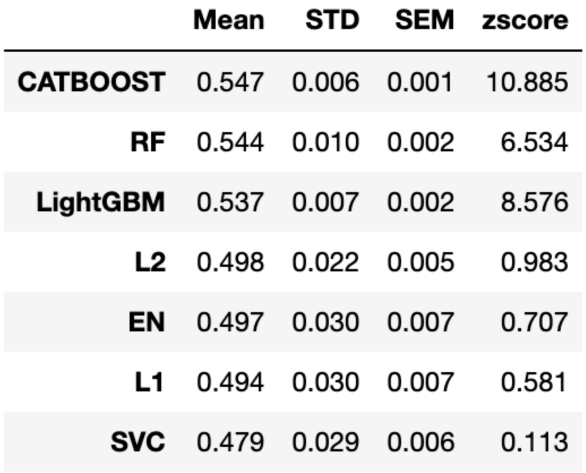
	Mean	STD	SEM	zscore
LightGBM	0.565	0.029	0.005	4.169
RF	0.559	0.019	0.003	6.054
CATBOOST	0.557	0.022	0.004	5.090
L1	0.513	0.042	0.007	1.654
L2	0.508	0.030	0.005	2.183
EN	0.503	0.045	0.007	1.356
SVC	0.483	0.038	0.006	1.037

Figure 7: The test scores’ overall mean and standard deviation.



	Mean	STD	SEM	zscore
LightGBM	0.592	0.011	0.003	16.086
RF	0.574	0.013	0.003	12.556
CATBOOST	0.567	0.028	0.006	5.635
L1	0.532	0.045	0.010	2.754
L2	0.518	0.033	0.007	3.248
EN	0.510	0.056	0.012	1.791
SVC	0.479	0.029	0.006	2.428

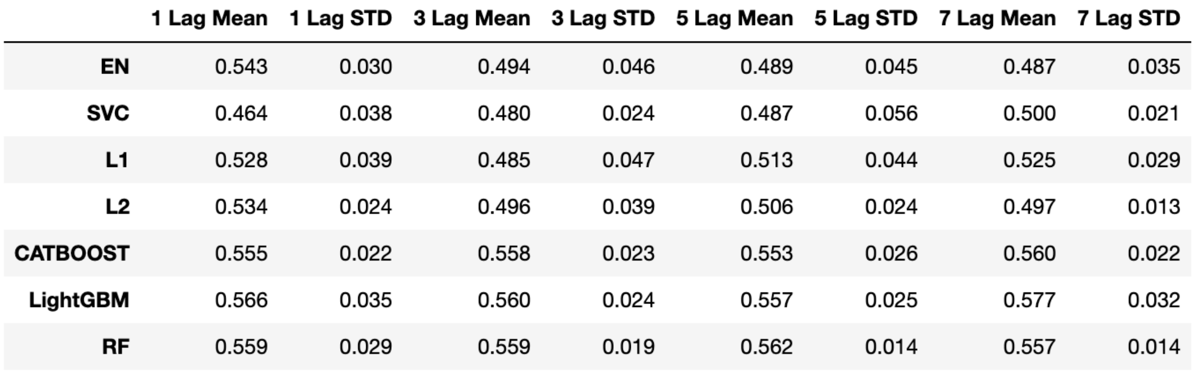
Figure 8: The mean and std for split-one.



	Mean	STD	SEM	zscore
CATBOOST	0.547	0.006	0.001	10.885
RF	0.544	0.010	0.002	6.534
LightGBM	0.537	0.007	0.002	8.576
L2	0.498	0.022	0.005	0.983
EN	0.497	0.030	0.007	0.707
L1	0.494	0.030	0.007	0.581
SVC	0.479	0.029	0.006	0.113

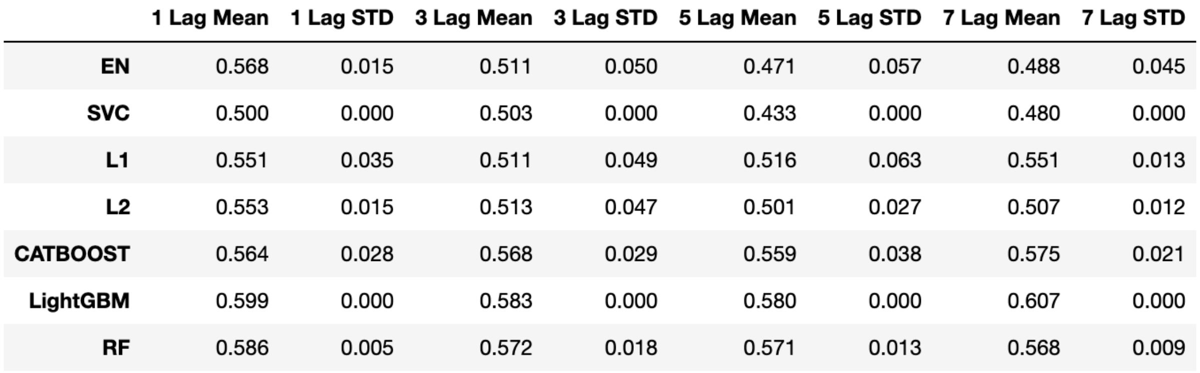
Figure 9: The mean and std for split-two.

I also computed the uncertain calculations for each lag.



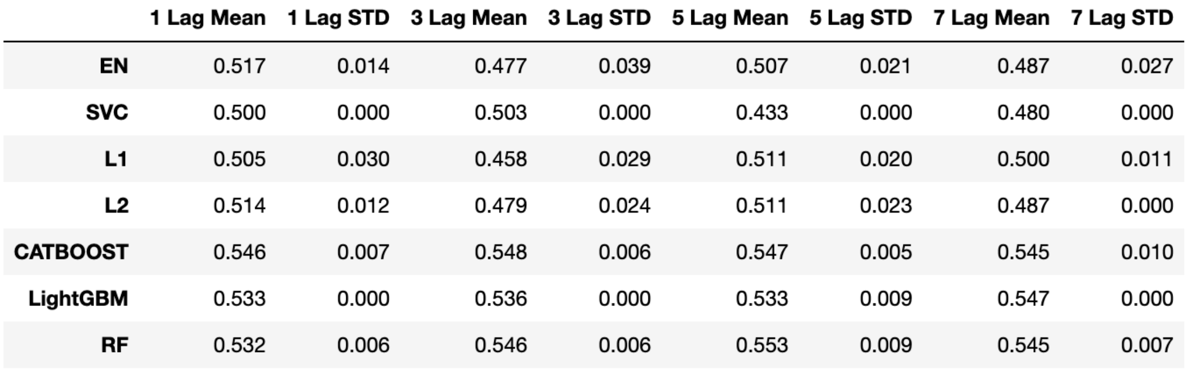
	1 Lag Mean	1 Lag STD	3 Lag Mean	3 Lag STD	5 Lag Mean	5 Lag STD	7 Lag Mean	7 Lag STD
EN	0.543	0.030	0.494	0.046	0.489	0.045	0.487	0.035
SVC	0.464	0.038	0.480	0.024	0.487	0.056	0.500	0.021
L1	0.528	0.039	0.485	0.047	0.513	0.044	0.525	0.029
L2	0.534	0.024	0.496	0.039	0.506	0.024	0.497	0.013
CATBOOST	0.555	0.022	0.558	0.023	0.553	0.026	0.560	0.022
LightGBM	0.566	0.035	0.560	0.024	0.557	0.025	0.577	0.032
RF	0.559	0.029	0.559	0.019	0.562	0.014	0.557	0.014

Figure 10: Overall std, mean, zscore of each lag for all models.



	1 Lag Mean	1 Lag STD	3 Lag Mean	3 Lag STD	5 Lag Mean	5 Lag STD	7 Lag Mean	7 Lag STD
EN	0.568	0.015	0.511	0.050	0.471	0.057	0.488	0.045
SVC	0.500	0.000	0.503	0.000	0.433	0.000	0.480	0.000
L1	0.551	0.035	0.511	0.049	0.516	0.063	0.551	0.013
L2	0.553	0.015	0.513	0.047	0.501	0.027	0.507	0.012
CATBOOST	0.564	0.028	0.568	0.029	0.559	0.038	0.575	0.021
LightGBM	0.599	0.000	0.583	0.000	0.580	0.000	0.607	0.000
RF	0.586	0.005	0.572	0.018	0.571	0.013	0.568	0.009

Figure 11: The first splits' std, mean, zscore of each lag for all models.

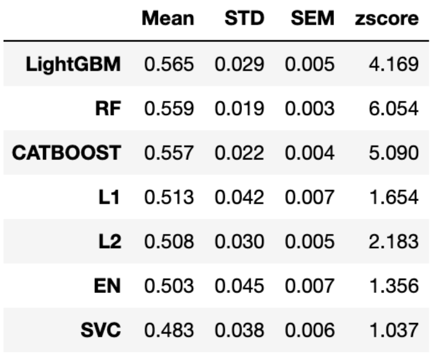


	1 Lag Mean	1 Lag STD	3 Lag Mean	3 Lag STD	5 Lag Mean	5 Lag STD	7 Lag Mean	7 Lag STD
EN	0.517	0.014	0.477	0.039	0.507	0.021	0.487	0.027
SVC	0.500	0.000	0.503	0.000	0.433	0.000	0.480	0.000
L1	0.505	0.030	0.458	0.029	0.511	0.020	0.500	0.011
L2	0.514	0.012	0.479	0.024	0.511	0.023	0.487	0.000
CATBOOST	0.546	0.007	0.548	0.006	0.547	0.005	0.545	0.010
LightGBM	0.533	0.000	0.536	0.000	0.533	0.009	0.547	0.000
RF	0.532	0.006	0.546	0.006	0.553	0.009	0.545	0.007

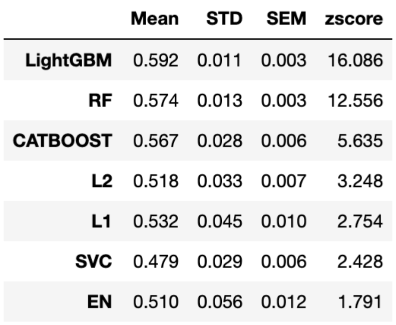
Figure 12: The second splits' std, mean, zscore of each lag for all models.

Results

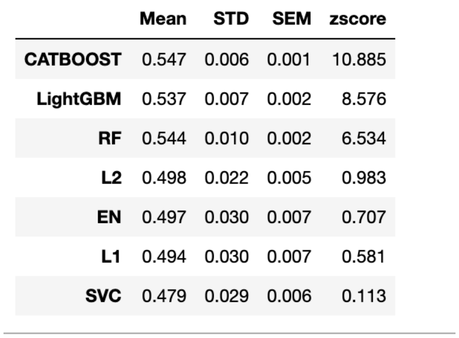
The overall baseline for each model was 0.4428, with a standard deviation of 0.0337. While random states did not affect the baseline, the different splits and lags did. The tables below summarize all zscores.



	Mean	STD	SEM	zscore
LightGBM	0.565	0.029	0.005	4.169
RF	0.559	0.019	0.003	6.054
CATBOOST	0.557	0.022	0.004	5.090
L1	0.513	0.042	0.007	1.654
L2	0.508	0.030	0.005	2.183
EN	0.503	0.045	0.007	1.356
SVC	0.483	0.038	0.006	1.037

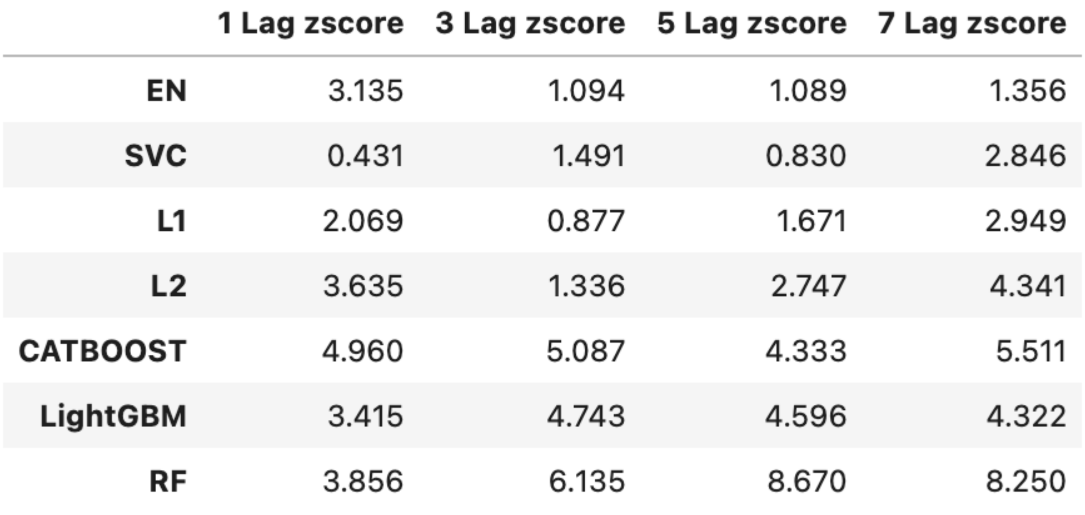


	Mean	STD	SEM	zscore
LightGBM	0.592	0.011	0.003	16.086
RF	0.574	0.013	0.003	12.556
CATBOOST	0.567	0.028	0.006	5.635
L2	0.518	0.033	0.007	3.248
L1	0.532	0.045	0.010	2.754
SVC	0.479	0.029	0.006	2.428
EN	0.510	0.056	0.012	1.791



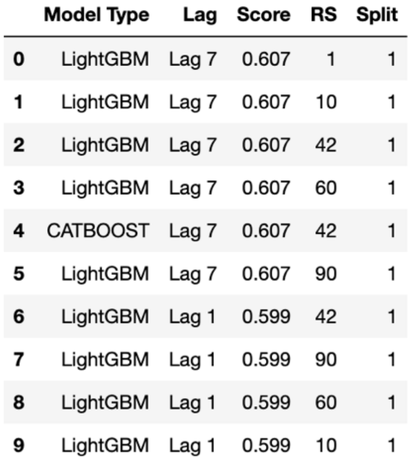
	Mean	STD	SEM	zscore
CATBOOST	0.547	0.006	0.001	10.885
LightGBM	0.537	0.007	0.002	8.576
RF	0.544	0.010	0.002	6.534
L2	0.498	0.022	0.005	0.983
EN	0.497	0.030	0.007	0.707
L1	0.494	0.030	0.007	0.581
SVC	0.479	0.029	0.006	0.113

Figure 13-15: Combined-split (left), first-split (middle), second-split (right). RF had the greatest zscore with a value of 6.054, followed by Catboost and LightGBM. * A combined-split means that all values are averaged, not just those specific to a particular split.

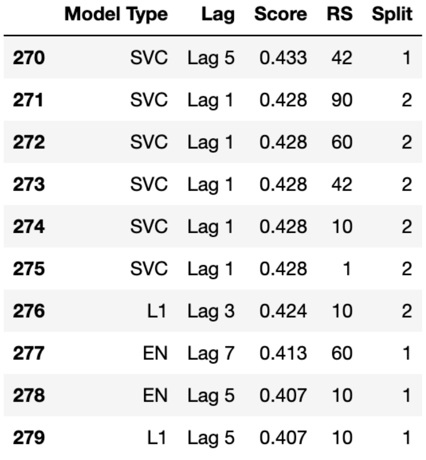


	1 Lag zscore	3 Lag zscore	5 Lag zscore	7 Lag zscore
EN	3.135	1.094	1.089	1.356
SVC	0.431	1.491	0.830	2.846
L1	2.069	0.877	1.671	2.949
L2	3.635	1.336	2.747	4.341
CATBOOST	4.960	5.087	4.333	5.511
LightGBM	3.415	4.743	4.596	4.322
RF	3.856	6.135	8.670	8.250

Figure 16: Overall RF has the highest zscores across the different lags, with CATBOOST and LighGBM coming in second and third, respectively. SVC performed the worst with the smallest zscores.



	Model Type	Lag	Score	RS	Split
0	LightGBM	Lag 7	0.607	1	1
1	LightGBM	Lag 7	0.607	10	1
2	LightGBM	Lag 7	0.607	42	1
3	LightGBM	Lag 7	0.607	60	1
4	CATBOOST	Lag 7	0.607	42	1
5	LightGBM	Lag 7	0.607	90	1
6	LightGBM	Lag 1	0.599	42	1
7	LightGBM	Lag 1	0.599	90	1
8	LightGBM	Lag 1	0.599	60	1
9	LightGBM	Lag 1	0.599	10	1



	Model Type	Lag	Score	RS	Split
270	SVC	Lag 5	0.433	42	1
271	SVC	Lag 1	0.428	90	2
272	SVC	Lag 1	0.428	60	2
273	SVC	Lag 1	0.428	42	2
274	SVC	Lag 1	0.428	10	2
275	SVC	Lag 1	0.428	1	2
276	L1	Lag 3	0.424	10	2
277	EN	Lag 7	0.413	60	1
278	EN	Lag 5	0.407	10	1
279	L1	Lag 5	0.407	10	1

Figures 17, 18: Top ten performing models (left) and the worst ten (right). LightGBM/CATBOOST were the highest-performing models, with top scores of 0.61. The worse was L1/EN, with scores of 0.406667.

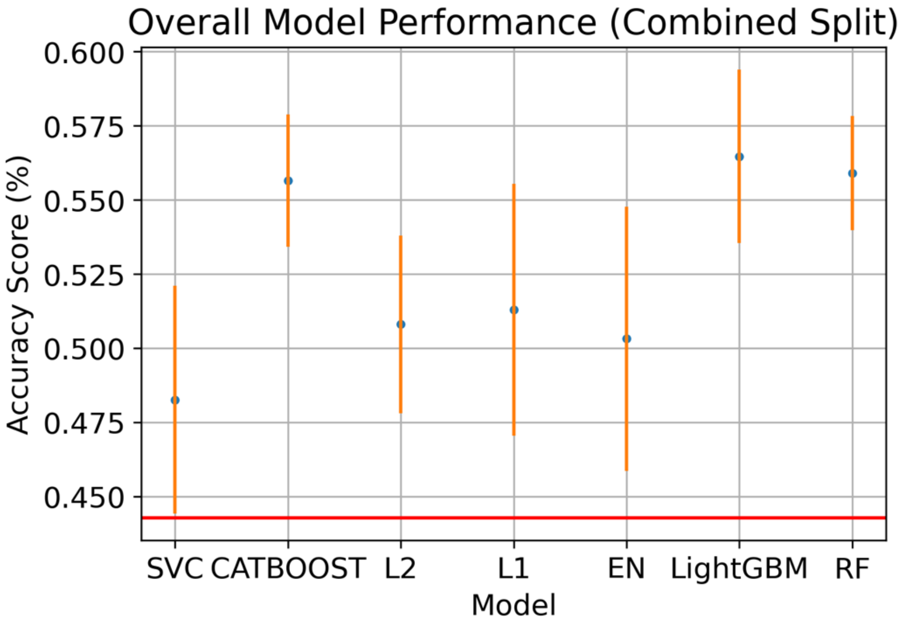


Figure 19: Each model's mean test score and standard deviation. The red line in all following plots is the baseline. LightGBM is the best performing, while RF has a smaller standard deviation.

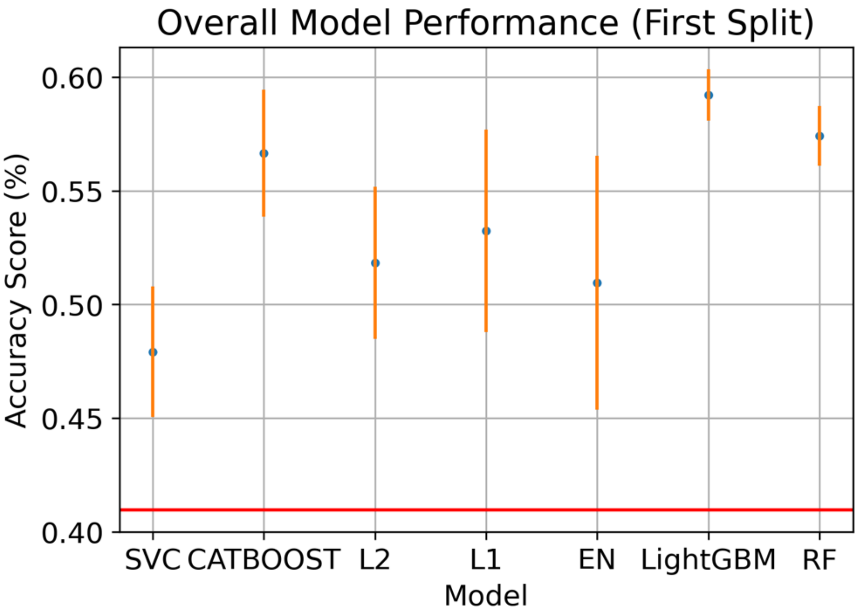


Figure 20: Each model’s mean test score and standard deviation at split-one. LightGBM is ahead of other models.

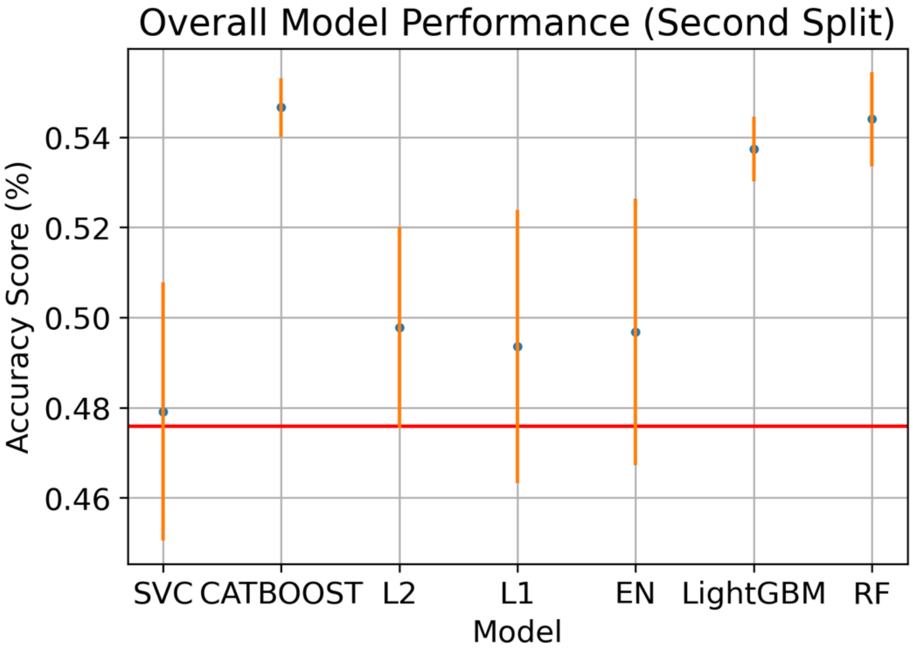


Figure 21: Each model’s mean test score and standard deviation at split-2. RF and CATBOOST do best.

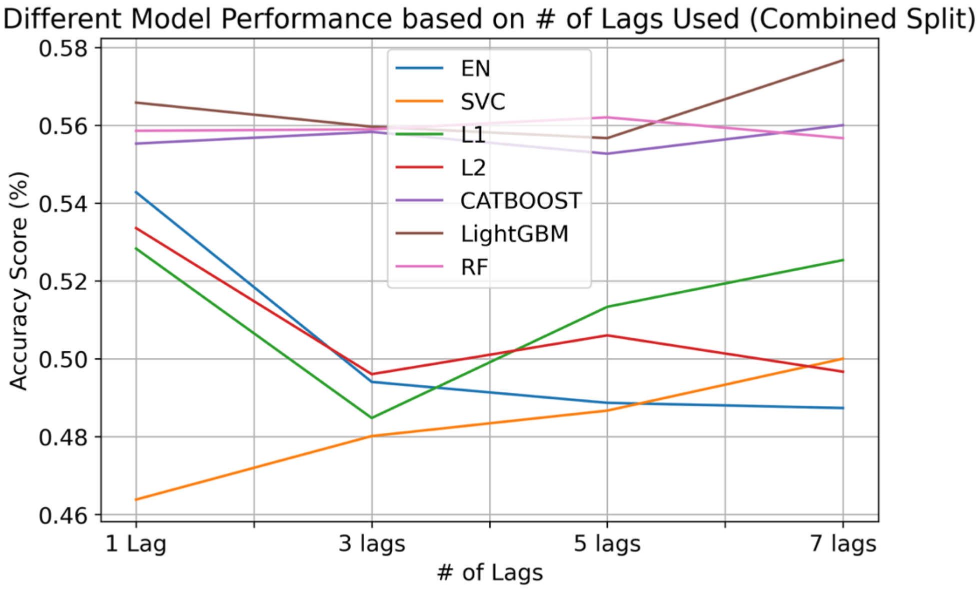


Figure 22: How different lags relate to performance. The overall trend is mixed. Three models (SVC, LightGBM, CATBOOST do better, while the rest do worse as lags increase). An upward trend from 5 to 7 lags may show that more lags could earn better performance.

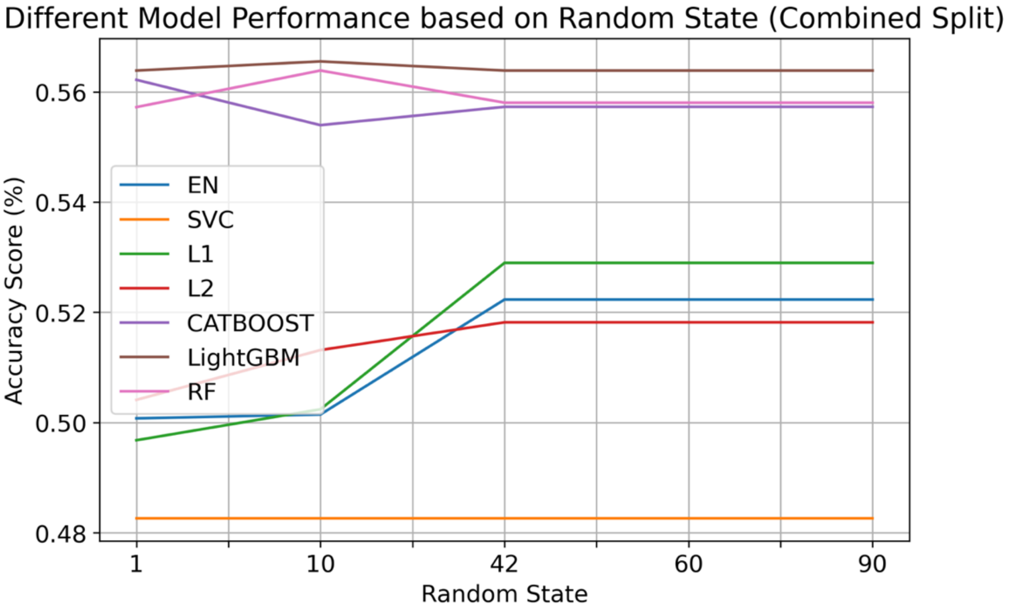


Figure 23: How the various random states affected the models' performance. SVC is horizontal due to deterministic nature. The others show a general trend of minor improvement from RS1 to RS90.

The model with the highest scores was LightGBM. It highest test-score was 0.600667 and is 20% greater than the baseline of 0.40667 for its respective lag and split. I used three methods to calculate feature importance: sklearn permutation importance (MDA), LightGBM's version of Gini (MDI), and Shap.

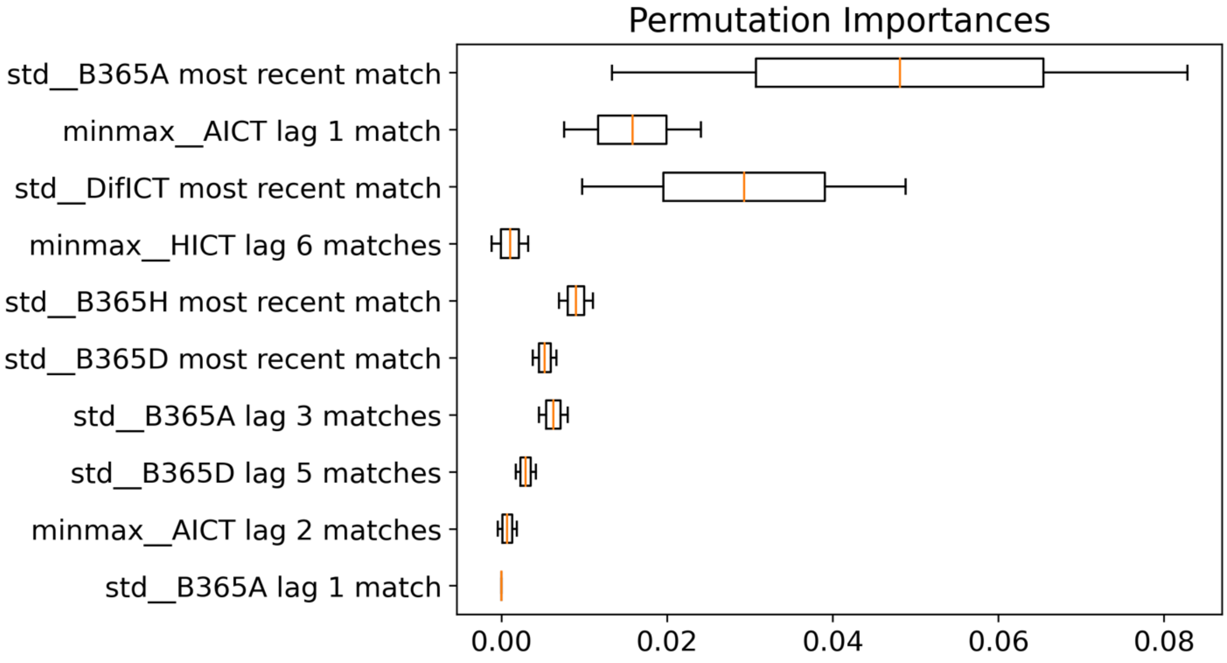


Figure 24: This graph makes use of MDA. This entails shuffling the characteristics N times and re-fitting the model to determine its relevance. Only seven features are particularly significant. The plot demonstrates that wagering on the away team has the biggest impact. This makes sense since the home team has a statistical edge over away teams; hence the strength or weakness of an away team affects its betting odds. The second is DifICt, which emphasizes team strength disparities. The third factor is AICT. This feature assesses the strength of the visiting team and relates it to the odds suggested by the first feature. The remaining features lose significance based on this graph.

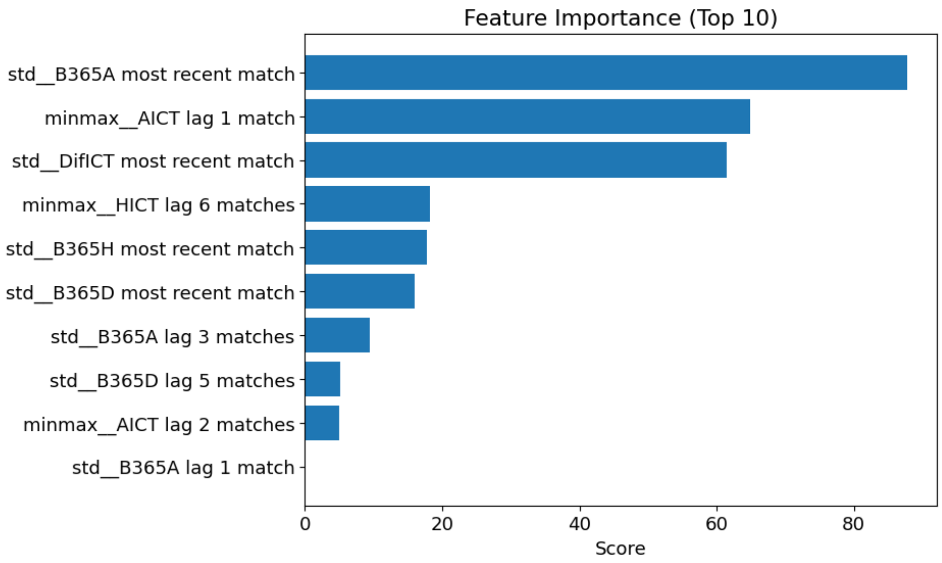


Figure 25: This feature importance is gain-based like MDI. Like the earlier plot, both methods detect the same features as the most important. However, the relative importance varies and order is different. This could result from how purity-based feature importance can be misleading for high cardinality features. In this plot, only nine features have any importance.

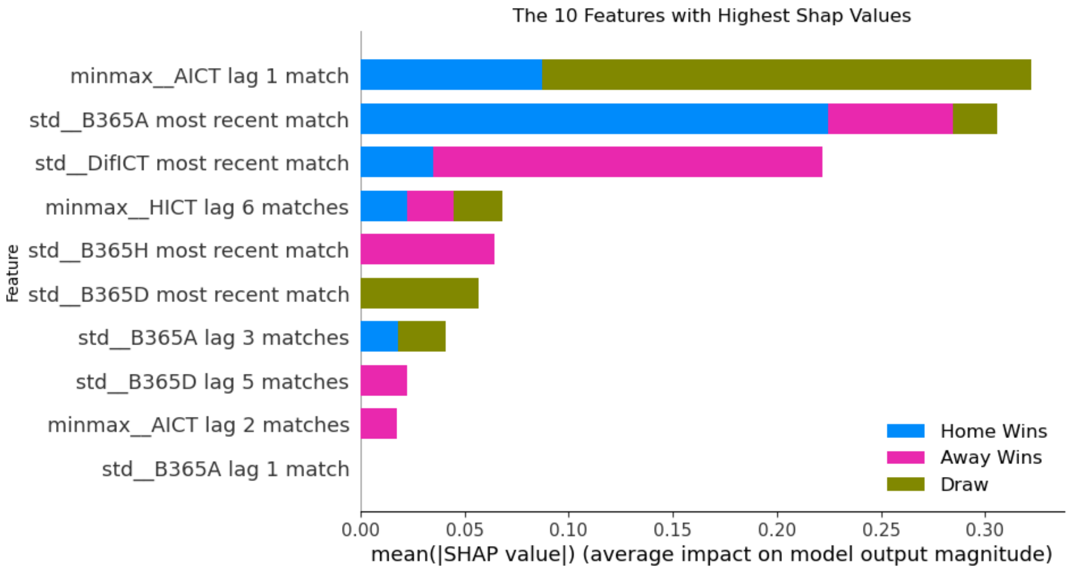


Figure 26: This figure is a Shap summary plot that shows feature importance from Shap values. Again, the same features appear but in different ordering. AICT Lag 1 is particularly predictive for draws, which makes sense as away team strength is the biggest indicator owing to home-field advantage. Other elements are minor. Nine features are important.

These three features' importance suggests that the other features are unimportant outside the 7-9 features. There are as many as forty-nine features, making 1/7th useful.

The three force plots below come from the best LightGBM and show the three different outcomes.

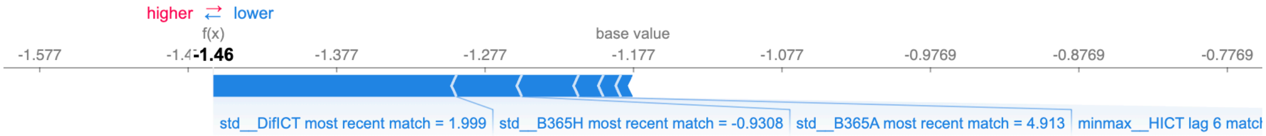


Figure 27: First is a loss with a base score of -1.777, and the predicted score is -1.46. District and B365H are the features that push the score lower and show the difference in strength between the teams. This pushes the prediction to the left. This force plot does not show a single value that actively pushes the prediction higher.

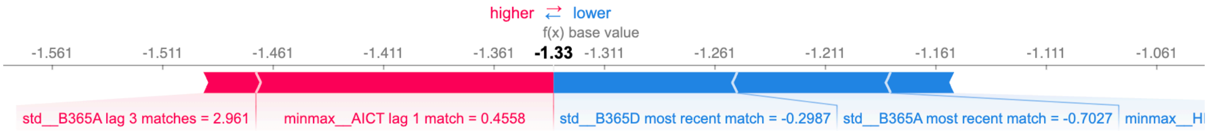


Figure 28: This force plot shows a draw with equal positive and negative features pushing the prediction to precisely the base value. The most important are AICT Lag one and B365D's most recent match.

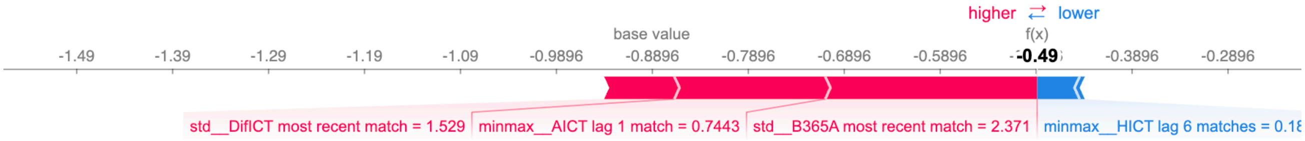


Figure 29: Lastly, this force plot shows a win where DifICT, AICT, and B365a all push the prediction higher than the base value of -0.8896. This indicates that these features have a positive relationship with the result for this data point, while HICT lag six matches have a negative relationship

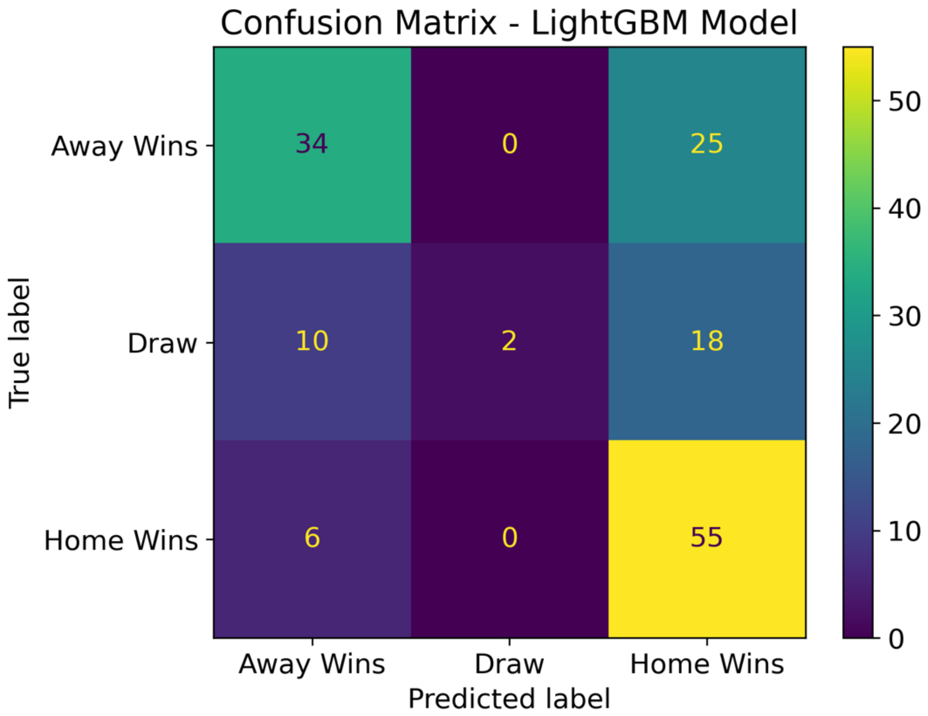


Figure 30: Above is LightGBM's Confusion Matrix. Predicting only two draws indicates the model will only forecast a draw if certain and the difficulty that draws pose to prediction. There are 55 TP, 43 FP, 34 TN, 16 FN. The model favors Home wins. The confusion matrix provides insight into a complicated aspect of the sport. Statistically, Draws should not exist. There is always a better team or a chance to score, but Draws account for ¼ of all outcomes and affect accuracy in all models. This inability to accurately predict draws shows football's unpredictability and beauty.

Outlook

The amount of data, computing power, and relevant features could have improved the model's success. A better dataset would hold more matches from the PL and five other important leagues. Then, one could use the past 20-40 seasons. This would equate to 1900 matches per annum over 20-40 years, making total data points equal to 38000-76000. An increase of 8215-16530%.

A more powerful computer would allow a more comprehensive range of parameters to tune quickly. This would allow for better tuning and better performance. Adding more relevant features like expected goals, expected away goals, average shots per game, and average saves per game would improve the accuracy by providing the computer with more statistics.

One could improve how the model determines accuracy. One could argue that predicting a Draw instead of a Win is better when the actual result is a Loss.

References

"Vaastav/Fantasy-Premier-League: Creates a .Csv File of All Players in the English Player League with Their Respective Team and Total Fantasy Points." Accessed October 21, 2022. https://github.com/vaastav/Fantasy-Premier-League.

"England Football Results Betting Odds | Premiership Results & Betting Odds." Accessed October 21, 2022. https://www.football-data.co.uk/englandm.php.

"FIFA 23 Complete Player Dataset [UPD:29/09/22]." Accessed October 21, 2022. https://www.kaggle.com/datasets/cashncarry/fifa-23-complete-player-dataset.

Ajgaonkar, Yash, Anagha Patil, Kunal Bhoyar, and Jenil Shah. "Prediction of Winning Team Using Machine Learning." *International Journal of Engineering Research & Technology* 9, no. 3 (February 22, 2021). https://doi.org/10.17577/IJERTCONV9IS03096.

Balawejder, Maciej. "Premier League Predictions Using Artificial Intelligence." *Nerd For Tech* (blog), September 20, 2022. https://medium.com/nerd-for-tech/premier-league-predictions-using-artificial-intelligence-7421dddc8778.

Harries, Owen. "Premier League Clubs RANKED by Popularity across Social Media." *Herald Wales* (blog), February 14, 2022. https://www.herald.wales/sport/football/premier-league-clubs-ranked-by-popularity-across-social-media/.

"IResearch | ASSESSMENT OF THE RELATIONSHIP BETWEEN SPORTS BETTING AND VIEWERSHIP OF ENGLISH PREMIERSHIP LEAGUE (EPL)." Accessed October 21, 2022. https://iresearchng.com/index.php/management/assessment-of-the-relationship-between-sports-betting-and-viewership-of-english-premiership-league-epl/index.html.

Change.org. "Sign the Petition." Accessed October 21, 2022. https://www.change.org/p/ban-anthony-taylor-from-officiating-chelsea-games.

"The Cost of Hosting a FIFA World Cup," Deccan Herald, September 25, 2022, https://www.deccanherald.com/sports/football/the-cost-of-hosting-a-fifa-world-cup-1148151.html.

"Fantasy Premier League, Official Fantasy Football Game of the Premier League," accessed October 21, 2022, https://fantasy.premierleague.com/.