Beating the Bookies

Harnessing Machine Learning and the Kelly Criterion for Premier League Betting Profits

Båth Viderström, Oscar
Sjöberg, Viktor

Supervisor: Peter Gustafsson

Master´s Thesis, 15 hp

Lund University: School of Economics and Management

Department of Statistics

22/05/2024

Abstract

In this study, we have explored the viability of using machine learning models to predict the outcomes of Premier League football matches. Especially we have evaluated the effectiveness with the Kelly Criterion as a betting strategy. We processed a comprehensive dataset to develop six distinct predictive models, each designed to forecast the probability of three possible match outcomes: a home win, a draw, or an away win. These models included various advanced machine learning techniques, and their performance was enhanced through hyperparameter tuning employing both grid search and randomized search methods.

Our methodology involved simulating an entire Premier League season, applying the Kelly Criterion to manage and allocate bets based on the probabilistic outputs of our models. The objective was to evaluate the potential returns on investment from each model. Among the models tested, the neural network emerged as the most successful, yielding a return of 35.48 times the initial bankroll. Other models demonstrated varied levels of success, illustrating the diverse potential of machine learning applications in sports betting.

The results of our simulations suggest that while machine learning can indeed be a powerful tool in sports betting, its efficiency depends on several factors. The quality of the input data and the level of sophistication of feature engineering might influence model performance. Our study highlights that further enhancements, such as incorporating more information and/or variables in our data and exploring alternative approaches to feature engineering, could improve predictive accuracy. Additionally, domain knowledge remains a critical component, suggesting that a hybrid approach combining data-driven techniques with expert insight may yield the most robust long-term betting strategies.

In conclusion, while our findings are promising, indicating that the application of machine learning and the Kelly Criterion can be profitable, they also underscore the need for ongoing refinement and the integration of comprehensive domain knowledge to fully capitalize on this approach in the competitive sports betting market.

Keywords: *Premier League, Betting, Machine learning, Neural Network, XGBoost, Random Forrest Classifier, Naive-bayes, Support Vector Machine, Logistic Regression, Kelly Criterion, Data Science.*

Acknowledgments

We would like to express our deepest gratitude to our supervisor, Peter Gustafsson, for his invaluable guidance and support throughout the course of this thesis. Peter has been more than a supervisor; he has been the best teacher we had during our time at the university. His door was always open, and he was always willing to help us, no matter what challenges we faced. His commitment and encouragement were instrumental in the completion of this work.

We also want to extend our heartfelt thanks to Professor Malgorzata Bogdan. Her insightful lectures and expertise in machine and deep learning have significantly enriched our knowledge and skills. She has been a great mentor and a constant source of support throughout our master's program and during the development of this thesis. Her willingness to assist and guide us has been invaluable.

Thank you both for your dedication, patience, and unwavering support. This thesis would not have been possible without your contributions.

Notation

ht	- Home Team
at	- Away team
s	- Season/year
md	- Match day
HW	- Home Win
D	- Draw
AW	- Away Win
CHG	- Cumulative home goals
CAG	- Cumulative away goals
CHW	- Cumulative home wins
CAW	- Cumulative away wins
CHD	- Cumulative home draws
CAD	- Cumulative away draws
$CHSOT$	- Cumulative home shots on target
$CASOT$	- Cumulative away shots on target
CHC	- Cumulative home corners
CAC	- Cumulative away corners
CHF	- Cumulative home fouls
CAF	- Cumulative away fouls
$CHYC$	- Cumulative home yellow cards
$CAYC$	- Cumulative away yellow cards
$CHRC$	- Cumulative home red cards
$CARC$	- Cumulative away red cards

Contents

1 Introduction

”In poker you never play your hand, you play the man across from you” This memorable and famous piece of wisdom, delivered by James Bond in the movie ”Casino Royale”, suggests that the game of poker is not solely about probabilities and mathematical odds. Indeed, poker is as much about understanding and predicting human behavior as it is about the hands dealt. A well-timed bluff, leveraging this very principle, can turn the weakest hand into the winning one. This psychological aspect of poker raises a fascinating question about its applicability to other competitive fields, particularly sports and the realm of sports betting.

Sports, much like poker, are not solely governed by the odds displayed on a betting slip. These odds, set by bookmakers, reflect a combination of statistical analysis, public perception, and historical data, so the betting companies can make the most money. However, is it possible for a discerned gambler, armed with a deeper understanding of the game, the players, and perhaps even the psychological dynamics at play, to consistently outsmart these odds? Is it possible to train a machine learning model to get all these valuable insights?

This thesis seeks to explore the intricate dance between statistical probability and human intuition in the context of betting. It will delve into whether a nuanced approach, one that goes beyond ”just looking at the odds”, can indeed provide a strategic edge in predicting outcomes. Through comprehensive analysis and examination of case studies, we will attempt to use our knowledge in machine learning and see if and how one can beat the odds.

1.1 Background

1.1.1 Premier League

As stated in in section 1.2 this thesis will focus on the English football division Premier League. The Premier League, officially founded on February 20, 1992, and since then represents the pinnacle of the English football. It was formed when the clubs in the First Division decided to break away from the Football League to take advantage of lucrative television rights deals. Initially comprised by 22 teams, it has since been reduced to 20 clubs to enhance competition and quality both domestically and on the international stage. This league operates on a system of promotion and relegation, which means the three teams that performed worst in the Premier League each year gets relegated to the second division. The three teams that performs the best in the second division gets promoted to the Premier League. (PremierLeague, 2024)

The Premier League quickly rose to become the most watched sports league globally, broadcast in 212 territories to an estimated 643 million homes, with a potential TV audience of 4.7 billion people. The competition is notable not just for its intense and unpredictable matches but also for its significant financial impact, with clubs benefiting from a share of billions in television rights deals. For example, the domestic rights for broadcasting 128 and 32 games were secured by Sky and BT Group, respectively, with deals amounting to billions of pounds and expected to increase in the coming years. (Wikipedia contributors, 2024c)

Throughout its history, the Premier League has seen 51 clubs compete, but only a select few having won the title. Manchester United leads with 13 championships, followed by Manchester City and others. The league has been a showcase for legendary players and memorable moments, contributing significantly to its standing as a global sporting phenomenon. (Wikipedia contributors, 2024c).

1.1.2 Betting

The evolution of betting and bookmaking has been deeply intertwined with the development of sports, notably the Premier League, where it adds another layer of engagement for fans. Beginning as informal wagers, the industry saw significant organization with the emergence of bookmakers like Ladbrokes in the 1886, evolving through the legalization of off-course betting in 1961, which paved the way for the proliferation of betting shops across the UK. This history has been marked by continuous adaptation to legal and technological changes, notably the shift to online platforms which drastically altered the landscape for traditional bookmakers. (Turcu et al., 2020).

In recent years, the relationship between the betting industry and the Premier League has become more formalized, with many clubs securing sponsorships and partnerships with betting companies. This relationship underscores the significant role that betting plays in football culture, though it also reflects the broader challenges that traditional betting shops face in the age of online gambling. With the advent of the digital era, bookmakers have been compelled to innovate to remain relevant, leading to a blend of the traditional betting shop experience with the convenience and breadth of online betting. (Casino.org, 2016).

Who set the odds

The two most usual way for bookmakers to set their odds is with *In-house Analysis* and *Third-party Services. In-house Analysis* is when the bookmakers develop their own algorithm to set odds. *Third-party Services* is when bookmakers pay another company to make the algorithm. One thing both methods have in common is *Dynamic Adjustment.* The odds need to change during the match depending on what is happening during and most importantly, *Market Influence.* Customer betting patterns also play a big role. If a lot of money is being wagered on a particular outcome, a company may adjust the odds to make other outcomes more attractive, balancing their risk and maximizing their gain. This means that the odds are not just based on statistics and algorithms but also on what humans think will happen. (McGrath and Pempus, 2023)

1.2 Problem statement & Thesis Objectives

This thesis will focus on the premier league, the highest division of English football. With a defined data set, several models will be created with an aim of predicting probabilities of different outcomes for a football match. These predictions for each probability of the outcome of a match: Home Win (HW), Draw (D) or Away Win (AW) will then be used as input to an algorithm which is based on the *Kelly Criterion theory*, and is explained in section 4.7. The evaluation metric for testing the models will be an implementation of the *Kelly Criterion theory.* The algorithm will be a guidance for which team one should bet on for each match and how much, depending on the relationship between odds set by bookmakers and the probabilities from our model to find the most "value" in bets. (Wikipedia contributors, 2024b) As discussed in section 1.1.2 the markets influence has a role in what odds will be available. This could be an advantage that our models could solely focus on statistics and not look at *which teams has the most fans who are willing to bet on them.* However, the human factor could also be a disadvantage since football is a psychological game and maybe, only humans understand it.

The gap that we are filling in the context of today's research is that this exact study has never been done before. Similar studies has been made, the closest we have found being that of Christoffersson, 2023. What distinguishes this study from Christoffersson's is our unique way of managing our data, the methods we are using (in particular the *Kelly Criterion*) and the data we both train and test our models on. By using this approach, we want to examine if this approach is a reliable way to make returns over the 2022-2023 premier league season.

Lastly, we are (probably) at the end of our academic journey and want to use as much as we can of what we have learned during our master's degree. Instead of tuning one model to perfection, we will create six unique models with different methods to show what we have learned and at the same time draw conclusion on which method/s works best for this particular data set.

Questions at Issue:

Question 1: Can we beat the bookmakers over the course of the 2022-2023 season?

Question 2: Is it possible in the long run, using the data we have and the models we have chosen, to make our algorithm-strategy a reliable source of income on betting in the premier league?

2 Previous Studies

The digital age has revolutionized the concept of prediction, significantly impacting revenue growth across various sectors, including sports. Sports predictions, often approached as a classification problem (win, lose, draw) benefit from structured experimental methods for optimal outcomes from datasets. These predictive models are pivotal for creating data products, aiding clubs in making informed decisions through recommendation systems. Extending beyond club-specific models, researchers aim to generalize these predictive frameworks for broader applicability across global sports clubs, enhancing sports analysis and decision-making processes.(Demmert, 1973)

The historical focus of sports analysis has been on ticket sales and fan attendance, key revenue sources for sports teams. Early empirical studies explored demand determinants, including local income, stadium age, team success and local market size. Subsequent econometric studies highlighted factors like recent team performance and star player impact on match attendance, distinguishing between controllable variables (e.g., opponent, match day, ticket price) and uncontrollable ones (e.g., weather conditions). (Noll, Coleman and Noll, 1974)

The pandemic shifted focus towards player performance and injury prevention, requiring digitized player histories. Studies utilized variables such as age, height, weight, and anthropometric features to predict player injury predisposition and physical performance, respectively (Rossi et al., 2018). Another study further examined factors affecting player substitution times and performance metrics like acceleration and energy expenditure. (Dijkhuis, Kempe and Lemmink, 2021)

Advancements in data accessibility have propelled football analysis, with football journals documenting match events being a primary data source. This has facilitated in-depth analyses of soccer at team and individual levels, despite challenges in assessing player performance quality due to inconsistent data handling. This evolving field continues to explore various dimensions, including ticket sales, fan attendance, player performance, and injury prevention, utilizing a wide range of variables and models for comprehensive sports analysis.(Bornn, Cervone and Fernandez, 2018)

A previous study by Christoffersson, 2023 was recently made that compared the accuracy of "machine learning models with the probabilities generated by sports betting companies", where a *Support Vector Machine* Support Vector Machine model performed the best with a 52.4 % accuracy whereas the betting companies had 40.4 %. The author considered the lowest betting odds set by the betting companies as the outcome their models deemed most probable.

3 Data

3.1 Dataset

The data is imported from *Kaggle* and consist of every premier league match and statistics since the 1993-1994 season. (Azarenka, 2024)

The raw data (before preprocessing) can be found in table 3.1.

Columns	**Descriptions**
Season	Which season the game is played
Date	Specific date the game is played
Time	The time of the date
HomeTeam	The team that plays at home for the match
AwayTeam	The team that plays away for the match
FullTimeHomeTeamGoals	How many goals the home team scored this game
FullTimeAwayTeamGoals	How many goals the away team scored this game
FullTimeResult	H for home team win, D for draw, A for away team win
HalfTimeHomeTeamGoals	How many goals the home team scored this game at half time
HalfTimeAwayTeamGoals	How many goals the away team scored this game at half time
HalfTimeResult	The result at half time
Referee	The referee of the game
HomeTeamShots	How many shots the home team took
AwayTeamShots	How many shots the away team took
HomeTeamShotsOnTarget	How many shots the home team took on target
AwayTeamShotsOnTarget	How many shots the away team took on target
HomeTeamCorners	How many corners the home team was awarded
AwayTeamCorners	How many corners the away team was awarded
HomeTeamFouls	How many fouls the home team committed
AwayTeamFouls	How many fouls the away team committed
HomeTeamYellowCards	How many yellow cards the home team committed
AwayTeamYellowCards	How many yellow cards the away team committed
HomeTeamRedCards	How many red cards the home team committed
AwayTeamRedCards	How many red cards the away team committed
MarketAvgHomeTeam	The average market odds on a home team win
MarketAvgDraw	The average market odds on a draw
MarketAvgAwayTeam	The average market odds on an away win

Table 3.1: Description of the data before preprocessing

3.2 Data Preprocessing & Feature engineering

First, we need to divide the data into training and a test set. This was done with a 80% to 20% ratio in favor for the training data. Both these samples will be used during the training process. At last we will exclude the latest season (2022-2023) of the premier league and use it as present observation. All of our final results will be based on this latest season, and from now on will be called *betting data*. In summary we will have 3 data samples, train data, test data and bet data.

Since we want to use data from earlier in the season to predict probabilities for match outcomes, we can't use data from the same game as we are predicting. To get around this problem the data gets cumulative summed after each round a team plays. For example at the start of each season every team has zero wins, goals, fouls, corners etc. But for each game that have been played these statistics changes and keeps getting added to until a new season begins. This is done using a -1 lag where we index the matches for each team through the date the match is played.

Football is constantly changing and for every season the team goes through a *Transfer Window*, which is a period were the team can sell and buy new players. This window is open during January and August and at the end of August a new season begins. These two factors might make the historical data less valuable for future predictions. If we would to use too much historical data, it could be bad for a model when the data is changed and restructured too much(McCausland, 2020). Because of this, we made a decision to structure our data in a cumulative way and reset it for every season.

However, we can extract some valuable information in historical data. We have created three variables (5GHW, 5GAW, 5GH, 5GA, 5GCH, 5GCA in table3.2) that contains some past observation that represents a team's current*form*. Form is how well the teams performs under a shorter time period. Measuring form in football can be an important factor in determining the current shape a team is in at the moment. We decided to strictly look at the last 5 games for this, since we believe it could be a reasonable number of games to indicate how well a team will perform before any given game.

There is no *Perfect Number of Games* to look at when calculating form. If you consider too few games then a really bad or good game could have a higher influence. With too many games the metric could be more evenly distributed but will not have as much valuable information since a team's form generally doesn't last very long. (Krishnan, 2023). In our case we have settled to look at five past games to decide form. Furthermore we will derive three variables as form from the past 5 games:

- Number of games won
- Goals scored
- Goals conceded

A well known fact in football is that it is a significant advantage to play at home rather than away. The advantage comes from a combination of psychological, supporting fans, field knowledge etc. (Wikipedia contributors, 2024a) This phenomenon is also visible in our data set, in Figure 3.1 the distribution of match outcome is visible and the most frequent outcome of a match is that the home team wins.

This advantage is the ground for how the data is processed, as shown in Figure 3.2 we split almost every column in two 2 columns depending on if a team plays home or away.

Columns	**Descriptions**
HomeTeam	Which team plays at home
AwayTeam	Which team does not play at home
Date	Specific date
HT	The team that plays at home for the match
AT	The team that plays away for the match
CHG	How many goals has the home team cumulative scored during the season
CAG	How many goals has the away team cumulative scored during the season
CHW	Cumulative home wins for the home team
CAW	Cumulative away wins for the away team
CHD	Cumulative draws for the home team
CAD	Cumulative draws for the away team
$CHSOT$	Cumulative shots on target for the home team
$CASOT$	Cumulative shots on target for the away team
CHC	Cumulative corners for the home team
CAC	Cumulative corners for the away team
CHF	Cumulative fouls for the home team
CAF	Cumulative fouls for the away team
$CHYC$	Cumulative yellow cards for the home team
$CAYC$	Cumulative yellow cards for the away team
$CHRC$	Cumulative red cards for the home team
$CARC$	Cumulative red cards for the away team
$5GHW$	Number of wins of the last 5 games played for the home team
$5GAW$	Number of wins of the last 5 games played for the Away team
$5GH$	Number of goals scored of the last 5 games played for the home team
$5GA$	Number of goals scored of the last 5 games played for the Away team
$5GCH$	Number of goals conceded of the last 5 games played for the home team
$5GCA$	Number of goals conceded of the last 5 games played for the away team

Table 3.2: Description of the data after prepossessing

Table 3.2 displays every variable that will be included in every model that later will be explained and built. Since each variable is quite unique some models will prefer one variable over the other. Instead of test every set of variable for each method a

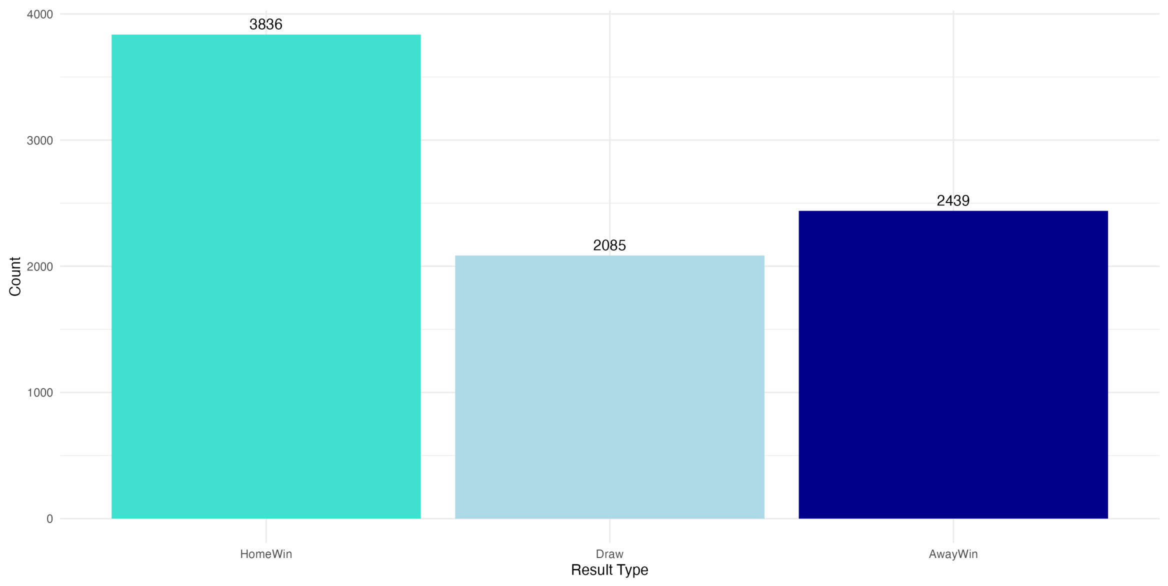


Figure 3.1: Distribution of match outcomes

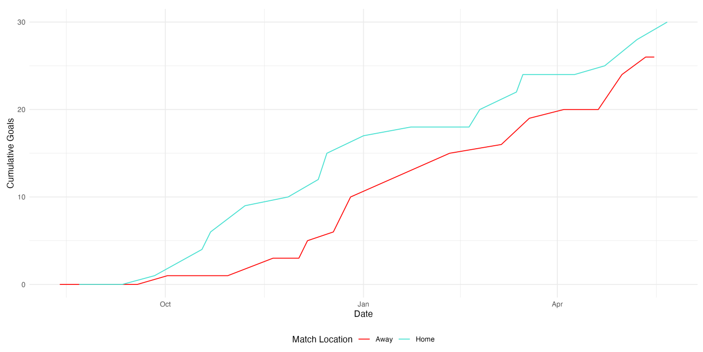


Figure 3.2: Arsenal Accumulated Goals (season 2021-2022)

regularization part will be added to some models and the tuning part will be further explained in chapter 5. Doing so we can use the same set of variable for each model but still let the models *choose* which variables that are the most important.

3.3 Descriptive Statistics

Figure 3.1 shows how the match outcome is distributed in our data set. The outcome which is the most frequent is home team win. In our data, 3836 of all 8360 matches ended in a home win, which is just around 46%.

Figure 3.2 shows how Arsenal goals evolves during the season. Arsenal scores more goals when they are playing at home then away. This does not only apply for Arsenal. The Figure also shows how our data is structured by always going up or sometimes stay the same if in this case arsenal does not score any goal.

What was done with arsenal goals over a season was also done on every variable displayed in table 3.2. This Figure shows how the variable conceded goals is distributed

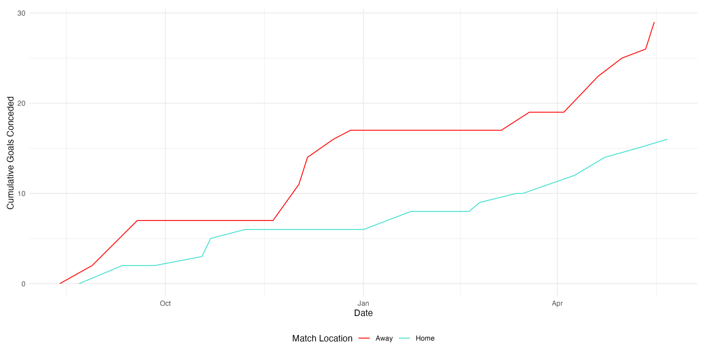


Figure 3.3: Arsenal Accumulated Goals Conceded (season 2021-2022)

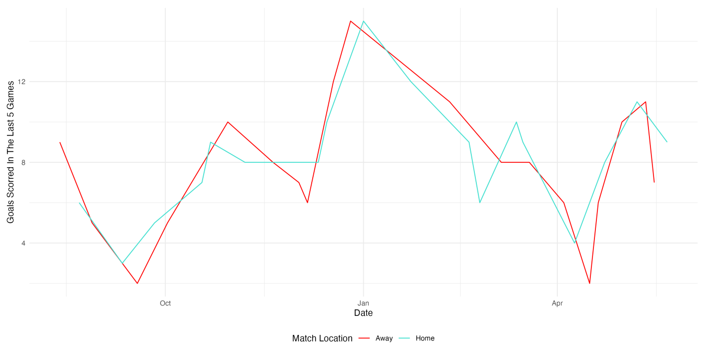


Figure 3.4: Arsenal Goals Scored in the Last 5 Games (season 2021-2022)

for home and away games played by the team arsenal.

As described in section 3.2 3 more variables were added to the data that only considered the last 5 games to include the teams' form in the model. In Figure 3.4 one of these variables is plotted over one season. The Figure shows Arsenals goals scored in the past 5 games. The Figure clearly shows Arsenal's form during the season, there is a clear correlation when they score lots of home goals they also score a lot of away goals indicating the teams high form during the seasons. Which also is clear when they are in a bad form.

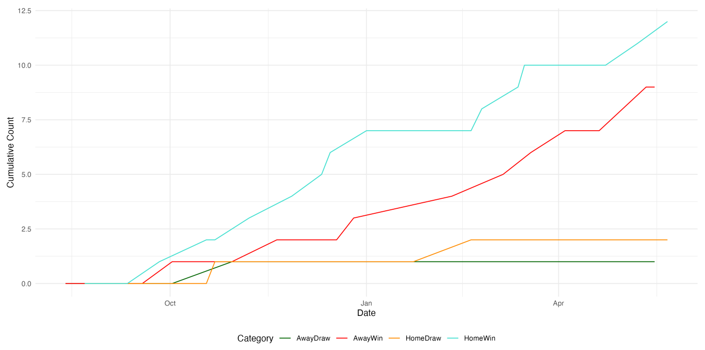


Figure 3.5: Arsenal Performance (season 2021-2022)

Betting and the Kelly Criterion

As described in section 1.2 we want to bet on premier league matches to try to make a profit. But how much return of *investment* does basic betting strategies yield.

Model	**Accuracy**	**Kelly Fraction**
Always Bet on home win	0.48	35.48
Always Bet on Away Win	0.23	-30.48
Always Bet on Draw	0.29	-57.04
Fixed Probabilities	NaN	3.37

Table 3.3: Basic Betting Strategies, Accuracy and Kelly Fraction

In table 3.3 the profit and accuracy is displayed depending on if one bets that: the home team wins, the away team wins or if the match ends in a draw. The accuracy is describing how many of the bets would have been correct. More specifically how many home wins there were this season if one choose the first basic betting strategy. The Kelly Fraction is calculated by setting your bankroll to x, then multiplying this number with the odds and adding the money won on the bet if you win or subtract the amount you played if the bet is lost. In the end the Kelly Fraction will show the rate of change of the initial bankroll x. Later in this thesis we will train models to calculate the probability of each outcome for each match. However, the fixed probabilities comes from historical match outcomes from all of our dataset which is displayed in Figure 3.1. These probabilities then gets fed through an algorithm named *Kelly Criterion Theory*, which will be be explained more thoroughly in section 4.7. In short terms, the algorithm puts the odds for a mach against the probabilities to tell you how much one should bet and on what outcome. Without any modeling it is a safe bet to always put money on the team that plays at home. It should be stated for the record, that the odds used is the market average before each game.

Outcome	**Mean Odds**
Home Win	2.87
Draw	4.22
Away Win	4.57

Table 3.4: Mean odds for each outcome for bet data

In table 3.4 and Figure 3.6 it is shown how odds during season 2022-2023 are distributed. Odds cant go under 1 otherwise the player could lose money independent of the outcome and the odds can go as high as possible.

The distribution for a home win odds is skewed to the right, with a concentration of lower odds (between 1 and 3) and a long tail extending towards higher odds. This suggests that the majority of games had relatively low odds for a home team win, indicating that home win were often considered probable. The distribution is positively skewed, possibly following a log-normal distribution.

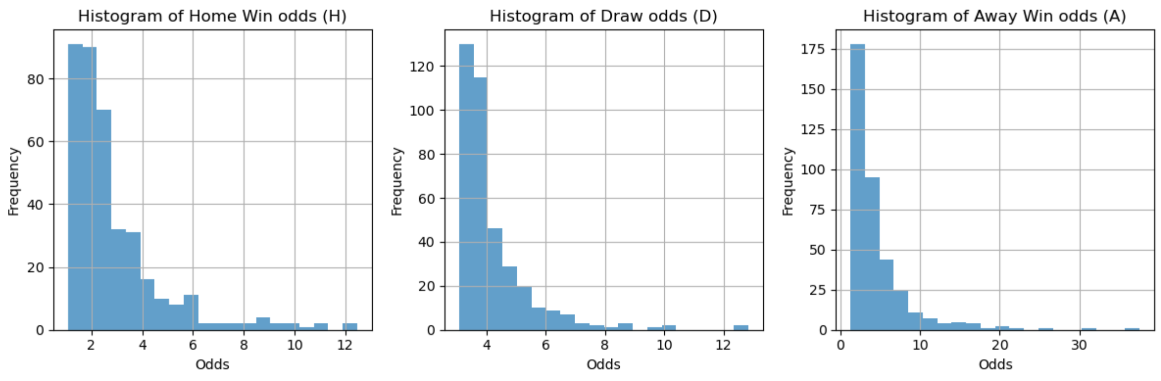


Figure 3.6: Distribution of odds in bet data

Similarly, the distribution of draw odds is also positively skewed, with a high concentration of odds between 2 and 4. The tail extends to higher values, suggesting that draws were generally considered as less likely outcomes. This distribution shape could also represent a log-normal or exponential distribution.

Lastly, the distribution of away win odds is the most skewed, with a very high frequency of low odds, specifically below 2.5. This shape indicates that away wins were generally less probable, with many games favoring the home team. The distribution is positively skewed with a heavy right tail, suggesting a log-normal or exponential distribution.

Position	Club	P	W	D	L	GF	GA	GD	P	Form
1	Man City	37	27	7	3	93	33	60	88	W W W W W
2	Arsenal	37	27	5	5	89	28	61	86	W W W W W
3	Liverpool	37	23	10	4	84	41	43	79	⊖ W L W W
4	Aston Villa	37	20	8	9	76	56	20	68	⊖ L L ⊖ W
5	Spurs	37	19	6	12	71	61	10	63	W L L L W

Table 3.5: Premier League Table (Top 5 teams date: 2024/05/17)

The official premiere league table that is updated after every match played is structured as table 3.5. We have had this structure as inspiration to for our data preprocessing.

- **P** - Games Played
- W - Won
- D - Draw
- L - Loss
- GF - Goals for (Goals Scored)
- GA - Goals Against (Goals Conceded)
- GD - Goal Difference
- P - Points
- W- Game Won
- ⊖- Game Drawn
- L- Game Lost

So for every input our models get will be similar to how a person sees the table before a game. The key difference is that we have spited the data in home and away games and we don't include points in our inputs.

4 Theory

4.1 Logistic regression

Logistic regression is a statistical technique that models the probability of a binary outcome as a function of one or more independent variables. This approach is particularly useful for situations where the response variable is categorical and has two outcomes.

The logistic model specifies the probability that the response variable Y equals one of the categories, which can be written as:

$$P(Y = 1|X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}} \tag{4.1}$$

The Model estimates the log odds, or logit, which is the natural logarithm of the odds that Y = 1

$$\log\left(\frac{P(Y = 1|X)}{1 - P(Y = 1|X)}\right) = \beta_0 + \beta_1 X \tag{4.2}$$

The linear relationship between the log odds and the predictor variable allows for the estimation of the coefficients β_0 and β_1 through maximum likelihood estimation. The resulting logistic function is an S-shaped curve that outputs a probability between 0 and 1 for any given value of X.

(James et al., 2020)

4.1.1 Multinomial Logistic Regression

Multinomial logistic regression extends the logistic regression model to multiclass problems where the outcome Y can take on more than two categories. It is particularly useful for predicting probabilities of the different possible outcomes in a classification problem.

In the multinomial logistic model, the probability of the response variable Y being in class k given predictor variables X is modeled as:

$$P(Y = k|X) = \frac{e^{\beta_{k0} + \beta_{k1} X_1 + \ldots + \beta_{kp} X_p}}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_{l1} X_1 + \ldots + \beta_{lp} X_p}} \tag{4.3}$$

For k = 1, ..., K-1, and for the baseline class K:

$$P(Y = K|X) = \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0}+\beta_{l1}X_1+\ldots+\beta_{lp}X_p}} \tag{4.4}$$

Each outcome class k has a separate linear predictor composed of the coefficients and features. These predictors calculate the log odds of being in class k versus the baseline class K. The coefficient are again estimated using the method of maximum likelihood.

(James et al., 2020)

Lasso (Least Absolute Shrinkage and Selection Operator) is a type of regularization technique that aims to enhance the predictive performance of models while simultaneously addressing overfitting issues. In the context of logistic regression, it involves adding a penalty term to the loss function, encouraging sparsity in the coefficient estimates.

To derive the logistic regression model, the mathematical objective is to minimize the negative log-likelihood of the data, which is typically expressed as:

$$L(\beta) = -\sum_{i=1}^{n} \left[y_i(\beta_0 + \beta_1 X_{i1} + \ldots + \beta_p X_{ip}) - \log(1 + e^{\beta_0+\beta_1 X_{i1}+\ldots+\beta_p X_{ip}}) \right] \tag{4.5}$$

In a lasso-regularized logistic regression model, an additional penalty term is added to this loss function to constrain the coefficients, preventing them from becoming too large:

$$L_\lambda(\beta) = -\sum_{i=1}^{n} \left[y_i(\beta_0 + \beta_1 X_{i1} + \ldots + \beta_p X_{ip}) - \log(1 + e^{\beta_0+\beta_1 X_{i1}+\ldots+\beta_p X_{ip}}) \right] + \lambda \sum_{j=1}^{p} |\beta_j| \tag{4.6}$$

Where:

- n: Number of data points.
- y_i: Response variable for the i-th data point.
- β_j: Coefficients of the model.
- λ: Regularization parameter controlling the strength of the penalty.

In multinomial logistic regression, the penalty term is applied to each set of coefficients corresponding to each class. The inclusion of the absolute value of coefficients in the penalty term forces some of them to shrink to zero, effectively performing feature selection.

Lasso regularization is especially useful in high-dimensional data where multicollinearity and overfitting are common concerns. By selecting a subset of relevant features, it enhances model interpretability and predictive performance.

(Ranstam and Cook, 2018)

4.2 Random Forest Classifier

Random forests are an ensemble learning method that combines the simplicity of decision trees with the power of averaging in order to improve predictive accuracy and control the amount of over-fitting. The random forest algorithm enhances the bagging technique by adding an extra layer of randomness to the model. The model randomizes the selection of features to split on at each node of each tree where it applies a bootstrap aggregating technique to develop multiple trees.

4.2.1 Overview of random forests

The random forest model operates by constructing a large number of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Mathematically, the prediction of a random forest is given by:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} T_b(x) \tag{4.7}$$

Where T_b is the b-th decision tree, and B i the total number of trees in the forest. Each tree is built on a bootstrapped sample of the data, and at each split, a random subset of m predictors is chosen from the full set of p predictors. This procedure decorrelates the trees and helps to reduce the variance of the model without a substantial increase in bias.

4.2.2 Out-of-Bag Error Estimation

Random forests utilize out-of-bag (OOB) error as a convenient substitute for cross-validation to estimate the test error. Each tree is constructed using a different bootstrap sample and not used in the construction of the k-th tree. These OOB cases are used to get a running unbiased estimate of the classification error as trees are added to the forest.

4.2.3 Variable Importance Measures

Random forests provide not only a prediction model but also a straightforward means for feature selection. In the context of random forests, variable importance is measured as the mean decrease in Gini impurity or the increase in accuracy when a variable is used for splitting. (Lindholm et al., 2022). The mean decrease in Gini impurity when variable X_j is used in a split across all trees can be expressed as:

$$\mathrm{VI}(X_j) = \frac{1}{B}\sum_{b=1}^{B}\left(\mathrm{Gini}_{\text{before}}(b) - \left(\frac{N_{\text{left}}}{N}\mathrm{Gini}_{\text{after, left}}(b) + \frac{N_{\text{right}}}{N}\mathrm{Gini}_{\text{after, right}}(b)\right)\right) \tag{4.8}$$

Here, $\mathrm{VI}(X_j)$ represents the variable importance of feature X_j, B is the total number of trees, $\mathrm{Gini}_{\text{before}}(b)$ is the Gini impurity of the parent node before splitting, $\mathrm{Gini}_{\text{after, left}}(b)$ and $\mathrm{Gini}_{\text{after, right}}(b)$ are the Gini impurities of the left and right child nodes after the split, and N_{left} and N_{right} are the number of samples in the left and right splits, with N being the total number of samples at the parent node. (James et al., 2020).

The Gini impurity for a node is typically calculated as:

$$\mathrm{Gini}(b) = 1 - \sum_{k=1}^{K} p_{k,b}^2 \tag{4.9}$$

Where $\mathrm{p}_{\mathrm{k,\ b}}$ is the proportion of class k observations at node b and K is the number of classes.

This equation sums the decreases in Gini impurity for each tree when feature X_j is used for splitting, then averages these decreases across all trees in the forest to obtain the overall variable importance score for X_j

(James et al., 2020).

4.3 XGBoost

A powerful machine learning algorithm which recently has gained popularity and attention is XGboost (eXtreme Gradient Boosting). This because of its capabilities in the effective way it can handle a number of different predictive modeling tasks. It's designed to be highly flexible, adaptable and efficient and the optimized distributed gradient boosting library emerges because of its ability to handle both high-dimensional and large-scale data. (Chen and Guestrin, 2016).

4.3.1 XGBoost's objective function

The objective function of XGboost, which consists of a regularization term and a differentiable convex loss function, is the core of XGBoost's methodology. Formally, it can be represented as:

$$L(\phi) = \sum l(\hat{y}_i, y_i) + \sum \Omega(f_k), \tag{4.10}$$

Where Ω is the penalty term that introduces regularization and penalizes the complexity of the model, and l represents the loss function that computes the difference between the predicted class $\hat{y}_i$ and the actual class y_i. In order to reduce overfitting, and enhance the model's ability to generalize well to new data, the two-component objective function mentioned in equation 4.10 is significant in reducing overfitting.

XGBoost applies a framework of gradient boosting, where sequentially new models are added in order to minimize and correct errors made by already existing models. The algorithm focuses on minimizing the loss function at each step, given by:

$$L(t) = \sum l(y_i, \hat{y}_i(t-1) + f_t(x_i)) + \Omega(f_t), \tag{4.11}$$

Where f_t is the function represented by the model added at the t-th iteration. This step-wise approach in the optimization process helps in progressively reducing residuals and to improve the accuracy of the model. (Chen and Guestrin, 2016).

4.3.2 System Optimization

XGBoost incorporates numerous system-level optimizations that boost its efficiency. Key enhancements such as approximate algorithms for tree learning and recognition of sparse data patterns contribute significantly to its design. These choices in system architecture allow XGBoost to effectively process extensive datasets, rendering it highly scalable and versatile across various computational settings. Chen and Guestrin, 2016.

The platform's effectiveness, ability to scale, and reliability position it as a tool for a variety of tasks in machine learning. XGBoost stands out due to its solid theoretical underpinnings combined with practical optimizations in system design, ensuring superior performance across diverse applications. Chen and Guestrin, 2016.

4.4 Support Vector Machines

4.4.1 SVM for binary classification

Support Vector Machines (SVMs) presents a framework for both binary and multiclass classification. By extending the concept of a linear classifier, SVMs aim to find an optimal hyperplane which maximizes the margin between two or more classes.

For linearly separable classes, the SVM seeks a decision boundary:

$$X^T\beta + \beta_0 = 0 \tag{4.12}$$

which maximizes the margin M, the distance between the closest points of the classes to the hyperplane, defined as:

$$M = \frac{1}{\|\beta\|} \tag{4.13}$$

The optimization problem for finding the maximum margin hyperplane is given by:

$$\max_{\beta,\beta_0} M \quad \text{subject to} \quad \frac{1}{\|\beta\|} y_i(\mathbf{x}_i^\top \beta + \beta_0) \geq M, \quad i = 1, \ldots, N \tag{4.14}$$

(Hastie, Tibshirani and Friedman, 2008).

Soft Margin and Slack Variables

In practice, classes may overlap. To handle cases where the classes can't be separated, SVM introduces slack variables ξ_i allowing some points to be within the margin or misclassified. The modified constraints are:

$$y_i(\mathbf{x}_i^\top \beta + \beta_0) \geq M(1 - \xi_i), \quad \xi_i \geq 0 \tag{4.15}$$

and the objective is to minimize:

$$\|\beta\| \quad \text{subject to} \quad \sum_{i=1}^{N} \xi_i \leq K \tag{4.16}$$

where K is a constant enclosing the sum of slacks.

(Hastie, Tibshirani and Friedman, 2008).

The Langrangian Dual Form

The problem's dual form introduces Lagrange multipliers a_i, offering a way to solve the SVM that only involves inner products:

$$L_D = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \tag{4.17}$$

Under this formulation, the decision function becomes:

$$\hat{G}(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i^\top \mathbf{x} + \beta_0\right) \tag{4.18}$$

SVM and Kernels

To capture non-linear relationships, SVMs use kernel functions K(x,x') to compute inner products in a transformed feature space. For example, the radial basis function (RBF) kernel is defined as:

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2) \tag{4.19}$$

allowing the SVM to construct non-linear boundaries in the original feature space.

(Hastie, Tibshirani and Friedman, 2008).

4.4.2 Multi-Class SVMs

By design, SVMs are mainly binary classifiers. However, multi-class classification tasks can be handled by employing different strategies. These include the popular one-vs-all (OvA) and one-vs-one (OvO) approaches. The OvA approach involves training a single classifier per class, with the samples of that class as positive samples and all other samples as negative. The OvO approach involves training a binary classifier for every pair of classes.

(Hastie, Tibshirani and Friedman, 2008).

One-vs-All

In the OvA approach, we train K separate binary classifiers $f_k(x)$ for K classes. For each classifier, the decision function determines whether an instance belongs to one class or to any of the other classes. The classifier $f_k(x)$ has the form:

$$f_k(\mathbf{x}) = \mathbf{x}^\top \beta_k + \beta_{0k} \tag{4.20}$$

For each k, the corresponding optimization problem with slack variables ξ_{ik}) :

$$\min_{\beta_k, \beta_{0k}} \frac{1}{2}||\beta_k||^2 + C\sum_{i=1}^{N} \xi_{ik} \tag{4.21}$$

is subject to

$$y_{ik}(\mathbf{x}_i^\top \beta_k + \beta_{0k}) \geq 1 - \xi_{ik}, \quad \xi_{ik} \geq 0, \quad \forall i \tag{4.22}$$

where y_{ik} = 1 if the true label y_i = k and y_{ik} = -1 otherwise, and C is the penalty parameter.

One-vs-One

In the OvO approach, K(K-1)/2 classifiers are built wher each onf is trained on data from two classes. For classes j and k, the classifier $f_{jk}(x)$ is trained on the subset of the training data that belongs to j and k. The classification decision is made based on a voting scheme where each classifier votes for one class. The final predicted class is the one with the most votes.

(Hastie, Tibshirani and Friedman, 2008).

Multi-Clas SVM Loss Function

The multi-class SVM can be formulated as a single optimization problem. The loss function is defined as:

$$L = \frac{1}{2}\sum_{k=1}^{K} ||\beta_k||^2 + C\sum_{i=1}^{N}\sum_{j\neq y_i} \max(0, 1 - \xi_{ij}) \tag{4.23}$$

where ξ_{ij} is the slack variable for the i-th instance when considering the margin relative to the j-th class.

(Hastie, Tibshirani and Friedman, 2008).

Kernal Extension for Multi-Class SVM

For non-linear decision boundaries, kernel functions are incorporated into multi-class SVMs:

$$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}') \tag{4.24}$$

where $\phi(\cdot)$ is a non-linear mapping to a high-dimensional space. The multi-class SVM classifier using the kernal trick can be written as:

$$f(\mathbf{x}) = \arg\max_k \left(\sum_{i=1}^{N} \alpha_{ik} y_{ik} K(\mathbf{x}_i, \mathbf{x}) + \beta_{0k} \right) \tag{4.25}$$

where a_{ik} are the Lagrange multipliers from the dual problem of the k-th binary classifier.

(Hastie, Tibshirani and Friedman, 2008).

4.5 Naive-Bayes

Naive Bayes (NB) classifiers are probabilistic models based on applying Bayes' theorem where assumptions about independence between the variables are applied. There is empirical evidence that NB classifiers are well-suited for high-dimensional datasets.

4.5.1 Bayes' Theorem Foundation

The core of the NB classifier is Bayes' theorem, which in the context of classification, is expressed as:

$$P(Y = k|X = x) = \frac{P(X = x|Y = k)P(Y = k)}{\sum_{l=1}^{K} P(X = x|Y = l)P(Y = l)} \tag{4.26}$$

where $P(Y = k|X = x)$ is the posterior probability that an instance x belongs to class k, $P(X = x|Y = k)$ is the likelihood of instance x given class k, and P(Y = k) is the prior probability of class k.

(James et al., 2020).

4.5.2 Independence Assumption

Naive Bayes simplifies the computation of the likelihood $P(X = x|Y = k)$ by assuming that the features in X are conditionally independent given the class label Y, such that:

$$P(X = x|Y = k) = \prod_{j=1}^{p} P(X_j = x_j|Y = k) \tag{4.27}$$

where X_j is the j-th feature of instance X, and p is the total number of features.

4.5.3 Parameter Estimation

For a dataset with N instances, K classes, and p features, the parameters of Naive Bayes are estimated as follows:

$$\hat{\pi}_k = \frac{N_k}{N} \tag{4.28}$$

where N_k is the number of instances in class k.

$$\hat{P}(X_j = x_j|Y = k) = \frac{1}{N_k} \sum_{i:y_i=k} I(x_{ij} = x_j) \tag{4.29}$$

where I is an indicator function that is 1 if $x_{ij} = x_j$ and 0 otherwise.

(James et al., 2020).

The final classification decision for an unseen instance x is made by choosing the class with the highest posterior probability:

$$\hat{y} = \arg\max_k \hat{P}(Y = k|X = x) = \arg\max_k \hat{\pi}_k \prod_{j=1}^{p} \hat{P}(X_j = x_j|Y = k) \tag{4.30}$$

(James et al., 2020).

4.6 Fully Connected Neural Network

A deep learning method that is used for many different applications is fully connected neural networks. Fully connected neural networks are what you can call ”structure agnostic”, meaning that they don’t make any special assumptions about the input. Because of this, fully connected neural networks are capable of learning any function from any kind of data whether they are images, videos, numeric and so on.

(Ramsundar and Zadeh, 2016).

A fully connected neural network, also known as a dense network, is composed of multiple layers where each neuron from one layer is connected to every neuron in the subsequent layer. The fundamental structure of a fully connected layer within such a network is depicted in Figure 4.1.

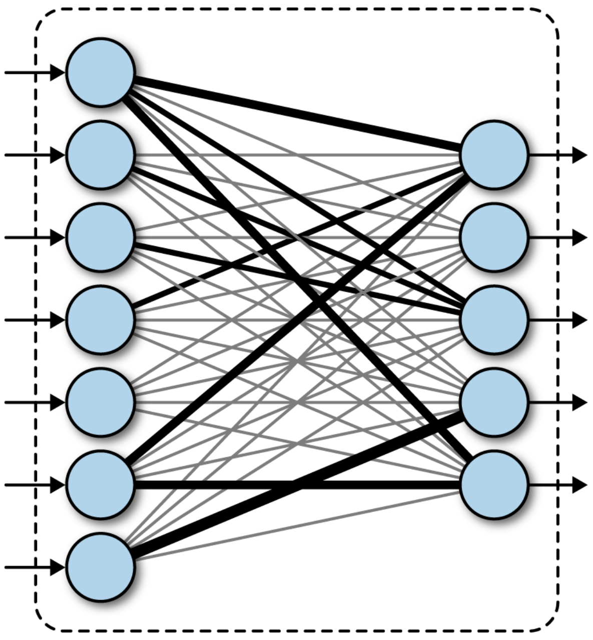


Figure 4.1: A fully connected layer in a deep neeural network

4.6.1 Mathematical Representation

Each layer in a fully connected neural network performs a linear transformation followed by a nonlinear activation. Let $\mathbf{x} \in R^m$ be the input vector to a fully connected layer. The output $\mathbf{y} \in R^n$ of this layer is given by:

$$y = \sigma(Wx + b) \tag{4.31}$$

Where:

- $\mathbf{W}$ represents the weight matrix of dimensions $n \times m$.
- $\mathbf{b}$ is the bias vector of length n.
- σ denotes the nonlinear activation function, applied element-wise. Common choices for σ include the sigmoid, ReLU (Rectified Linear Unit), and tanh functions.

(Ramsundar and Zadeh, 2016).

4.6.2 Neurons in Networks

Historically, the nodes in these networks are referred to as ”neurons”. This terminology comes from an analogy to neurons in the biological brain, as initially suggested by early neural network models proposed by Warren S. McCullogh and Walter Pitts in the 1940s. These mathematical models were designed to mimic the binary output of biological neurons. (Ramsundar and Zadeh, 2016).

4.6.3 Weight Regularization

To prevent overfitting and improve the generalization of the model, weight regularization techniques are applied. These techniques modify the loss function used to train the network by adding a penalty term related to the magnitude of the weights:

$$\mathcal{L}'(\mathbf{x}, \mathbf{y}) = \mathcal{L}(\mathbf{x}, \mathbf{y}) + \alpha\|\theta\|_p \tag{4.32}$$

where $\mathcal{L}$ is the original loss function, θ represents the weights of the network, α is a tuning parameter, and $|\cdot|_p$ is a norm representing either L1 or L2 regularization:

- L1 Regularization (Lasso): $|\theta|1 = \sum i = 1^N |\theta_i|$
- L2 Regularization (Ridge): $|\theta|2^2 = \sum i = 1^N \theta_i^2$

(Ramsundar and Zadeh, 2016).

4.6.4 Training Techniques

Minibatching

In practice, especially with large datasets, the training data is divided into small batches. This approach, known as minibatching, not only facilitates training on limited memory resources but also introduces noise into the gradient descent process, which helps in escaping local minima. (Ramsundar and Zadeh, 2016).

Learning Rates

The learning rate is a critical hyperparameter that influences how much the weights are adjusted during training. Modern optimizers like Adam automatically adjust the learning rate during training, which helps in stabilizing the training process. (Ramsundar and Zadeh, 2016).

4.7 Kelly Criterion Theory

The Kelly Criterion is a mathematical formula used to determine the optimal size of a series of bets. Developed by John L. Kelly Jr. in 1956, it originates from information theory, specifically the concept of maximizing the rate of wealth growth. The Kelly Criterion has found applications in gambling, investing, and risk management due to its effectiveness in balancing the trade-off between risk and reward. The Kelly criterion formula is is written as:

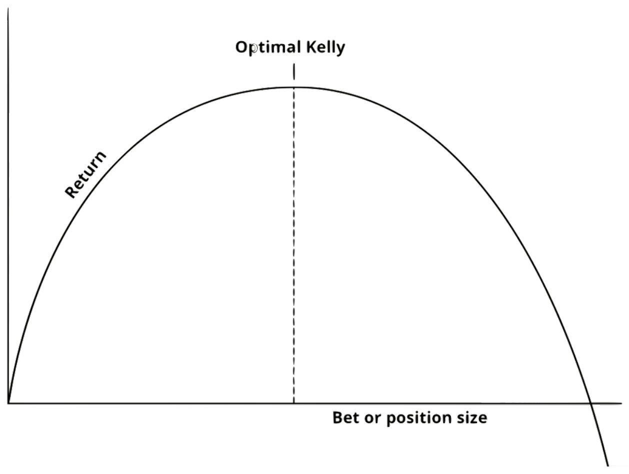


Figure 4.2: The Theoretical Kelly Index

$$b = \frac{(\text{odds} * \text{probability}) - 1}{\text{odds} - 1} \tag{4.33}$$

- b is the fraction of the bankroll to wager
- odds are the betting odds, given by market average
- probability are the probabilities calculated by our model(s)

The bankroll is the initial money one would like to gamble with for every bet the formula calculates how much of the bankroll should be put into that bet. This approach makes every win and loss small and over a long period of time generate a steady and reliable source of income.

Figure 4.2 illustrates the relationship between the bet size or position size and the return. The curve reaches its peak at the "Optimal Kelly" point, indicating the bet size that maximizes expected return.

To the left of this peak, smaller bets result in suboptimal growth, while to the right, larger bets increase risk disproportionately, potentially leading to significant losses. This balance ensures that while maximizing returns, risk exposure remains controlled, making it a powerful strategy for both gambling and investment decisions.

(Poundstone, 2010)

4.8 Grid & Random Search

When building a machine learning model, tuning hyperparameters is crucial for optimizing the performance of algorithms. Two popular methods for this are Grid Search and Random Search.

4.8.1 Grid Search

methodically explores a predefined subset of the hyperparameter. It involves choosing a grid of hyperparameter values and evaluating their performance. The grid contains what hyperparmeters the search should try and which values for each hyperparameter. This approach ensures a check of each combination within the grid, which can be set when dealing with real or unbounded parameters. And returns the set of hyperparmaters that yielded the highest choosen evaluation metric e.g. accuracy. Grid search is especially beneficial when hyperparameter interactions are significant since it evaluates all possible combinations.

$$I = P^N \tag{4.34}$$

- I is the number of iterations the search will do
- P is the predefined number of values each hyperparameter can take
- N is the number of hyperparameters

4.8.2 Random Search

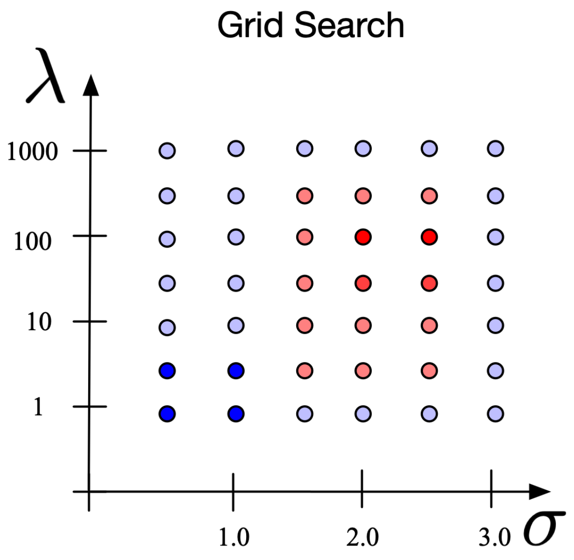
on the other hand, selects hyperparameter values randomly within the defined search space. This method does not try every possible combination but samples them stochastically. It is particularly effective when the parameter space is large or when only a few hyperparameters significantly influence the algorithm's performance. Random search can be applied to discrete, continuous, and mixed spaces. When executing random search one also predefined the number of iterations that should be conducted.

4.8.3 Grid vs Random Search

As illustrated in Figure 4.3, the primary difference between Grid Search and Random Search is in the pattern of their search. The Figure only displays if two hyperparameters are tested, in our case we will have more and therefore the Figure explaining our search will be in a higher dimensions.

In "Grid Search" part of Figure 4.3, the search points are structured and align along grid lines, covering the space uniformly. This systematic approach can be parallelized easily since each parameter combination is independent. However, it might become inefficient in high-dimensional spaces where many combinations yield similar results or are irrelevant.

In contrast, the "Random Search" part in Figure 4.3 shows a scattered pattern of points, indicating the random selection of hyperparameters. This randomness allows the method to explore a wider range of values with fewer iterations. As a result, random search can often find better parameters faster than grid search, particularly in cases where the optimal settings are sparsely located within the search space.



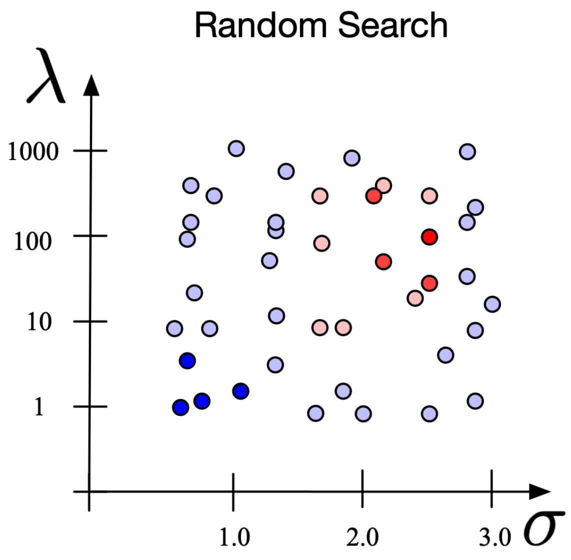


Figure 4.3: Grid vs Random Search

In summary, while Grid Search guarantees that every possible combination within the grid is evaluated, Random Search provides a more dynamic exploration of the search space, which can lead to quicker discoveries of optimal solutions, particularly in complex parameter spaces.

(Liashchynskyi and Liashchynskyi, 2019)

5 Method

5.1 Data extraction

As previously mentioned in section 3, the process of extracting the data was not the most complicated. In fact, we solely had to download a CSV file to extract the data.

5.2 Data preprocessing & feature engineering

Given our data structure, we had to preprocess some variables and perform feature engineering in order for us to build a model that could use the historic data to make predictions for our probabilities. In section 3.2 we delved into the variables we created. In that section we mention that we use cumulative statistics that for each game gets updated and reset by the end of the season. These cumulative statistics is also divided into two groups, home statistics and away statistics. This data preprocessing is done with the r library *dplyr*. With dplyr it is possible to group the data by one selected variable, in our case season. Further, from match statistics create a new variable with the cumulative sum for each game round by mutating the cumulative sum using *lag*. This is done to every variable that include any match statistics from table 3.1

Before the models where created we had to make a change in the test set. A team named *Nottingham Forrest* has never played in the premier league before. This mean that every model wont recognize the level for home/ away team that is *Nottingham Forrest*. What we did is that we gave *Nottingham Forrest* the coefficient of the team that came last in the prior season. The reasoning behind this change is that the best team in the second division (EFL championship) should be the closest team in regarding performance as the worst team in the premier league.

5.3 Model building & tuning

Every model is built in a unique way to try to maximize the methods advantages. During the training process we try to maximize the methods capabilities as much as possible given our computational power and time.

5.3.1 Fixed Probabilities

Rather than building a model to try to predict a match outcome there are some more unsophisticated betting methods are: pure feeling, always bet on the lowest respectively highest odds or only bet for one outcome e.g. Home Win. In this project there will be one unsophisticated method as a base method or benchmark. The model that will be presented will just be by looking at the past match outcome. As presented in figure 3.1 around 46% of all matches we have looked at have been a home win, 25% of the matches has concluded in a draw and 29% of the matches has ended in a Away win. These statistics has been set as fixed for every match in the 2022-2023 season, followed by applying the Kelly criterion algorithm.

5.3.2 Logistic regression

The first model we built was a multinomial regression where variable selection was performed using LASSO, meaning that our α=1. We did this because we wanted a starting point and to see if our model could make any sense of the data.

5.3.3 Random Forest

For our Random Forest model, we also started with a base model. However, we decided that we were going to tune it in order to optimize the model's performance. We used grid search for the tuning.

Parameter grid

The parameters we tuned, including the grids, can be seen in table 5.1. We performed the grid search with 5 cross validations.

Parameters	**Grid**
Max depth	40, 50, 60, 70, 80, 90, 100
Max features	2, 1, 3, 4, 5
Min samples leaf	7, 6, 5, 8, 9, 10, 11
Min samples split	14, 16, 12, 10, 18
Number of estimators	300, 600, 1000, 1100, 1300, 1500

Table 5.1: The grid of parameters searched for Random Forest

5.3.4 XGBoost

For XGBoost, due to the computational cost we experienced using the Grid Search for our Random Forest model, we used randomized search instead. We did this because it took approximately 10 hours to run the code using grid search. The Randomized search was performed using 10 early stopping rounds, "mlogloss" as our evaluation metric and with no verbose. We iterated the randomized search 20 times using 5 cross validations. The parameters that was configurated in our Randomized Search can be found in table 5.2.

Parameters	**Grid**
Number of estimators	Random integer between 50 and 150
Learning rate	0.05, 0.1
Max depth	Random integer between 3 and 7
Minimum child weight	Random integer between 1 and 3
Subsample	0.7, 0.8, 0.9
Gamma	0.1, 0.2
Alpha (Regularization)	0, 1e-2
Lambda (Regularization)	1, 1e-2

Table 5.2: The grid of parameters of our randomized searched for XGBoost

5.3.5 Support Vector Machines

For SVM the same applies as for the previous methods. We performed a random search where we had to limit the hyperparameters even more because of computational cost. For example, we only did 5 iterations and 3 CV folds. The parameters that was configurated in our Randomized Search can be found in table 5.3.

Parameters	**Grid**
C (Regularization parameter)	Exponential distributed with scale 10
Kernel	Linear, rbf
Gamma	'scale'

Table 5.3: The grid of parameters of our randomized searched for XGBoost

5.3.6 Naive-Bayes

For our Naive-Bayes model, we performed a grid search with 5 cross validations. Our only parameter was variable smoothing, where we generated 100 values between 10^0 and 10^{-9} on a logarithmic scale.

5.3.7 Neural Network

Below is the detailed architecture of the neural network used in our experiments:

height**Layer (type)**	**Output Shape**	**Param #**
InputLayer	(None, 31)	0
Dense (ReLU)	(None, 64)	2048
Dropout (0.5)	(None, 64)	0
Dense (ReLU)	(None, 32)	2080
Dropout (0.3)	(None, 32)	0
Dense (ReLU)	(None, 16)	528
Dense (Softmax)	(None, 3)	51

Params

- Total params: 4,707
- Trainable params: 4,707
- Non-trainable params: 0

Compilation Parameters

- **Optimizer:** Adam (learning rate: 1.0×10^{-5}, clipvalue: 1.0)
- **Loss function:** Categorical Crossentropy
- **Metrics:** Accuracy

Regularization and Convergence

- **Early Stopping:** Monitored on validation loss, patience: 5, restore best weights: True

When a data point is fed into the neural network, it first enters the input layer, which has 31 features. From there, the data flows through several layers, each designed to transform the input in a specific way to learn patterns from the data. The first transformation occurs in a dense layer with 64 neurons, using the ReLU activation function to introduce non-linearity, helping the network learn complex patterns. This layer's weights are initialized using the GlorotUniform method, ensuring optimal initial weights.

After the first dense layer, a dropout layer randomly sets 50% of the neuron outputs to zero during training, which helps prevent overfitting by ensuring that no single neuron becomes too influential on the outcome. The process repeats through another dense layer and dropout layer sequence, further refining the network's ability to generalize.

The data continues to a final dense layer of 16 neurons, again using ReLU, before reaching the output layer. The output layer has 3 neurons (corresponding to three classes) and uses the softmax activation function to output probabilities of each class, indicating the network's prediction.

The architecture was developed through a trial and error approach, relying on qualitative guesses about what configurations might yield good performance. By iteratively adjusting the layers, neuron counts, and dropout rates, we tried to optimized the network to balance learning capacity and generalization to unseen data.

5.4 Kelly Criterion

The Kelly Criterion is a formula used to determine the optimal size of a series of bets. It maximizes the expected logarithm of wealth and is given by equation 4.33.

The odds are gather from the data set made by Azarenka, 2024 and is the market average from the top bookmakers in the UK before the game starts, because of the *Dynamic Adjustment* discussed in section 1.1.2. The *Dynamic Adjustment* is not included in our models, all of the bets are placed before the match begins.

5.4.1 Algorithm Implementation

We implement the Kelly Criterion in Python and apply it to a DataFrame containing betting odds and predicted probabilities for different outcomes.

The Kelly criterion algorithm calculates the optimal fraction of the bankroll to wager using the Kelly Criterion. It takes the probability of winning (`prob`) and the odds (`odds`) as inputs and returns the fraction to bet.

Then `calculate_bets` function applies the Kelly Criterion to each possible outcome (Away win, Draw, Home win) and selects the outcome with the highest positive Kelly fraction. The chosen outcome, the fraction to bet, and the expected profit are then calculated. If the actual outcome matches the chosen bet, the profit is calculated as the fraction times the net odds; otherwise, the profit is negative, representing the loss of the bet fraction.

Using the Kelly Criterion to determine bet sizes allows for a disciplined and mathematically sound approach to betting. By integrating machine learning models to predict outcome probabilities, this method aims to maximize long-term growth of the bankroll.

6 Results

6.1 Comparison

Model	Training Accuracy	Kelly Fraction
Logistic Regression	0.47	3.54
Neural Network	0.46	35.48
Random Forest Classifier	0.45	-0.08
XGBoost	0.42	-0.16
Support Vector Machine	0.46	3.85
Naive-Bayes	0.40	2.43

Table 6.1: Model performance

Table 6.1 shows how well the different model preformed in terms of the match outcome classification accuracy and the profit or loss expressed as the fraction from the Kelly criterion. The accuracy for the different models are quite similar, the lowest scoring model is Naive-Bayes with an accuracy of 40%, compared to The logistic regression obtaining the highest accuracy of 47%. In section 1.2 the stated objective was not to obtain the highest accuracy, but yield the highest profit. The profit and loss column displayed as a fraction from the Kelly criterion algorithm shows that the neural network model made the most money from our betting data. With a fraction of 35.48, which means if someone bets one Swedish crown on each of the 380 matches in the betting data, they would have made 35.48 Swedish crowns as a profit. Two models lead to a loss after all of the bets were placed, Random Forest Classifier and XGBosst.Navie-Bayes had the lowest accuracy but it still yield a profit in the end, illustrating that accuracy is not the most important metric for this particularly problem.

When we compare the results to the basic betting strategies presented in table 3.3, no model came close to the loss given by always betting on a draw or away win. The logistic regression and neural network got the same result as always betting on win and fixed probabilities.

6.2 Multinomial Logistic Regression

Outcome	Mean Probability
Home Win	0.46
Draw	0.25
Away Win	0.29

Table 6.2: Mean probabilities for each outcome for Logistic Regression

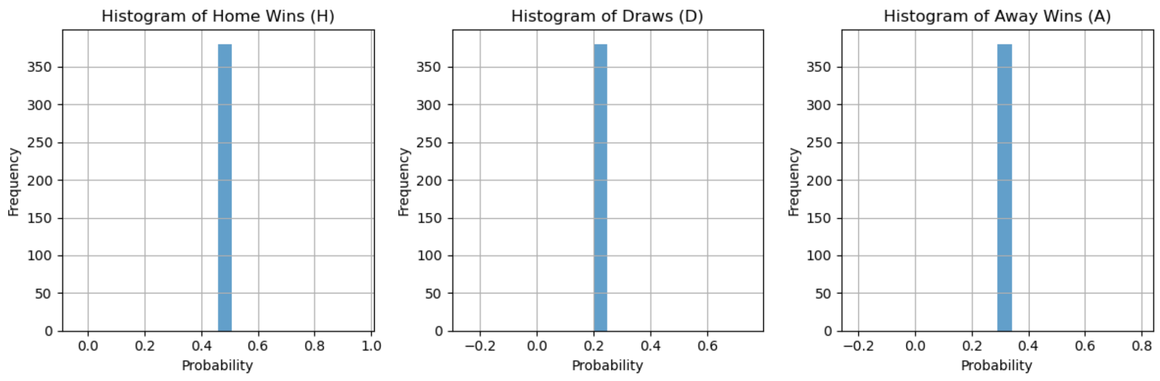


Figure 6.1: Distribution of outcomes for the Logistic Regression

As shown in Figure 6.1 the logistic regression sets all the probabilities for each outcome to the same value for each of the 380 matches in our bet data. This also mean that in table 6.2 the *Mean Probability* is the value that each outcome has, and there is no spread or variance. The mean probability is very close to the match outcome distribution in our training and test data.

In the end the logistic regression yielded a profit of 3.54 times the wagered money. Since the probabilities are the same for every outcome for each match the only different is the size of the bets, which is depended on the odds.

6.3 Random Forest Classifier

Outcome	Mean Probability
Home Win	0.43
Draw	0.26
Away Win	0.31

Table 6.3: Mean probabilities for each outcome for Random Forest

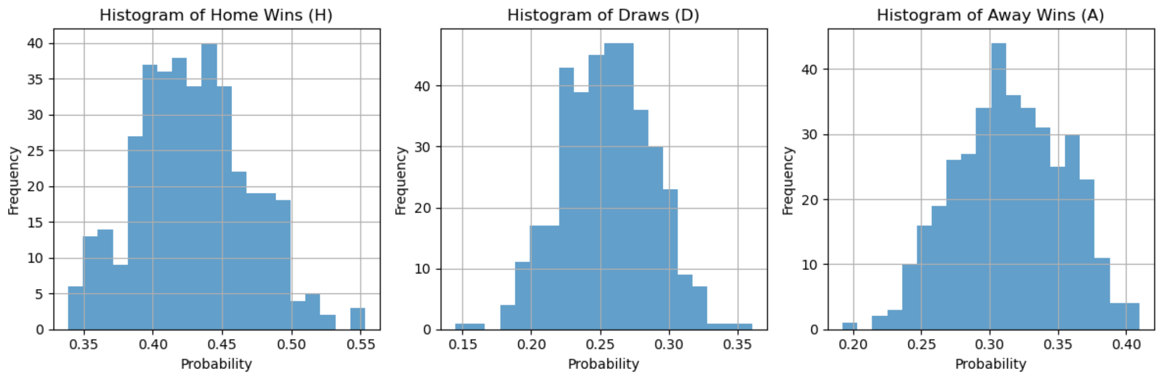


Figure 6.2: Distribution of outcomes for the Random Forest

Our Random Forest Classifier had a little wider spread than the logistic regression. The mean probability for a home win was around 43 %, draw 26 % and away win 31 %, as can be seen in table 6.3. In Figure 6.2, we can see the distribution of the different probabilities for each game. We can see that the lowest probability for a home win is about 35 %, and the highest being about 55 %. For draws the interval was between approximately 15 % and 35 %, and for away wins between 20 % and 40 %. This model seems that take more information into account, although it didn't make any returns but lost 8 % of the original bankroll, as can be seen in table 6.1. The distributions also looks rather normal.

6.4 XGBoost

Outcome	Mean Probability
Home Win	0.44
Draw	0.25
Away Win	0.31

Table 6.4: Mean probabilities for each outcome for XGBoost

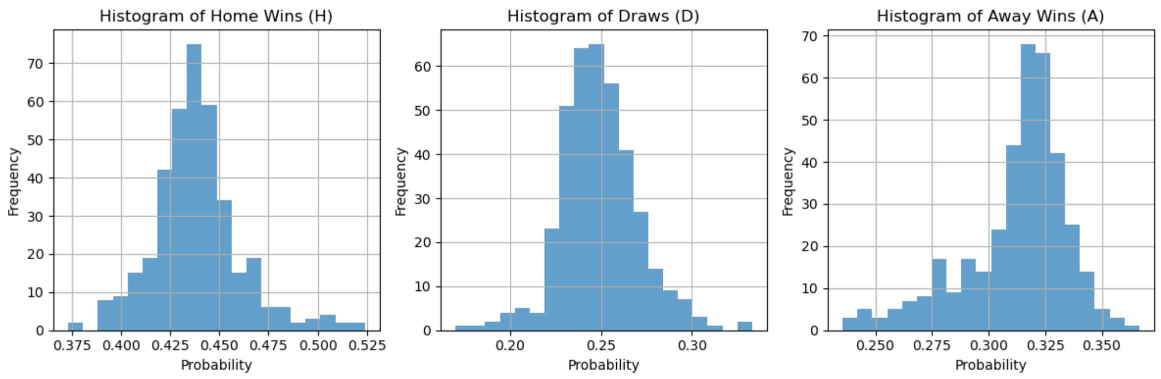


Figure 6.3: Distribution of outcomes for the XGBoost

Our XGBoost model had similar mean probabilities for each outcome. Namely 44 % for home win, 25 % for draw and 31 % for away win, which can be found in table 6.4. If we take a look at the distribution of the probabilities, we can also see that the interval as well as the distribution is pretty close to those of the Random Forest. In table 6.3 it can be noted that XGBoost has an interval between 37.5 and 52.5 % for home win, 15 and 35 % for draws, and 17.5 and 37.5 % for Away wins. These also looks rather normally distributed. As can be seen in table 6.1, the model also made losses of 16 % of our bankroll.

6.5 Support Vector Machine

Outcome	**Mean Probability**
Home Win	0.46
Draw	0.25
Away Win	0.29

Table 6.5: Mean probabilities for each outcome for SVM

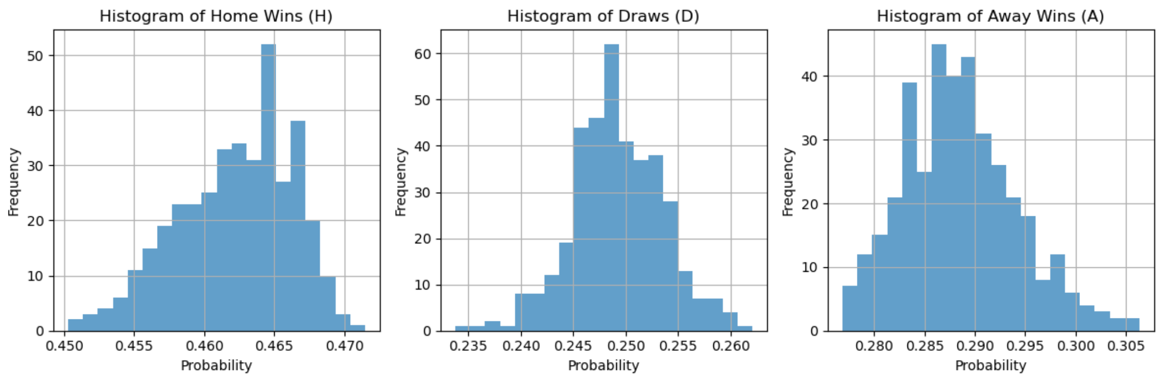


Figure 6.4: Distribution of outcomes for the SVM

The mean probabilities of our SVM model are also similar to both the XGBoost and Random Forest. SVM had a mean probability of 46 % for home win, 25 % for draw, and 29 % for away win, as per table 6.7. In Figure 6.4, the interval of the distribution of the probability for each outcome is narrow. A home win lies somewhere between 45 and 47 %, draw 23.5 and 26 % and away win 28 and 30.5 %. In table 6.1 we can see that SVM did make a return with a 3.85 multiplier of our bankroll

6.6 Neural Network

Outcome	Mean Probability
Home Win	1.00
Draw	0.00
Away Win	0.00

Table 6.6: Mean probabilities for each outcome for Neural Network

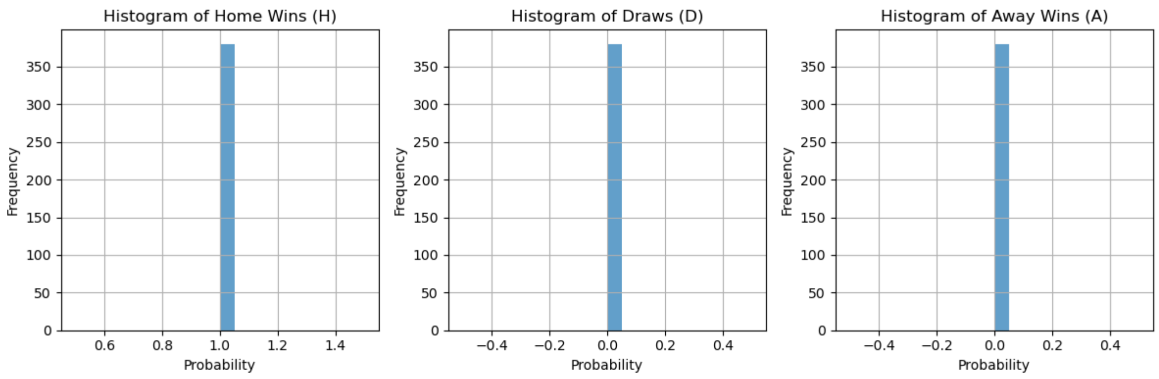


Figure 6.5: Distribution of outcomes for the Neural Network

Our neural network on the other hand set the probability of each game as a certain (100 %) home win. Although (or because of) this unsophisticated nature, it managed to make a return with a 35.48 multiplier on our initial bankroll. When the probability of a outcome is 100% the Kelly criterion will always lead to a bet fraction of 100%.

6.7 Naive Bayes

Outcome	Mean Probability
Home Win	0.45
Draw	0.25
Away Win	0.30

Table 6.7: Mean probabilities for each outcome for Naive-Bayes

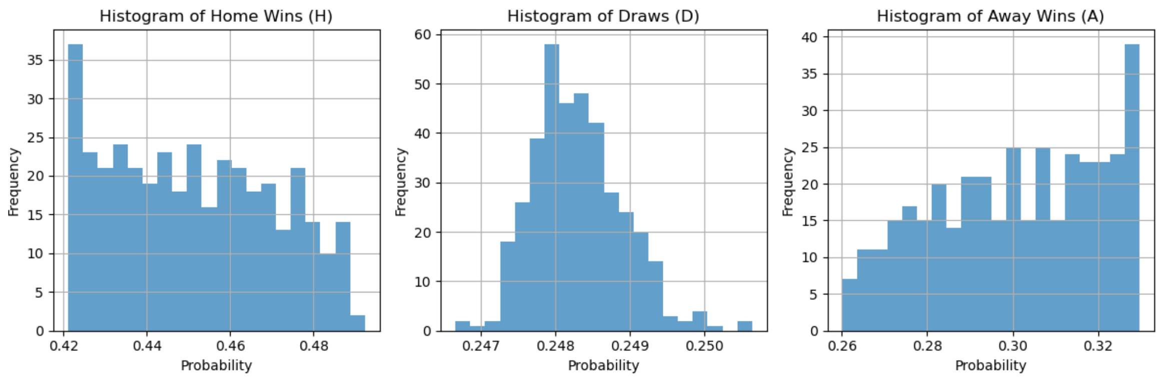


Figure 6.6: Distribution of outcomes for the Naive-Bayes

Our Naive Bayes model had a mean probability of 45 % on home win, 25 % on draw and 30 % on away win. The distribution ranges from 42 to 49 % for home win, 24.7 to 25.2 % for draw and 26 to 33 % for away win. Compared to home and away win, draw only has a 0.5 % range of different probabilities whilst it is 6 % for home win and 7 % for away win.

6.8 Kelly Criterion

As stated in section 4.7 the Kelly criterion tries to find the optimal betting strategy based on odds and probabilities. In the same section Figure 4.2 shows the theoretical Kelly index. In comparison to our empirical index illustrated in Figure 6.7 it is clear that the empirical index dose not match the theoretical index perfect but comes quite close to having the same shape. 6.7 is the index for fixed probabilities/ the probabilities generated by the logistic regression.

Figure 6.8 illustrates the empirical Kelly index for the neural network. In comparison to 4.2 the neural network does not follow the theoretical index. However, it resulted in the highest profit.

The rest of the models dose not follow the theoretical index and are included in appendix B

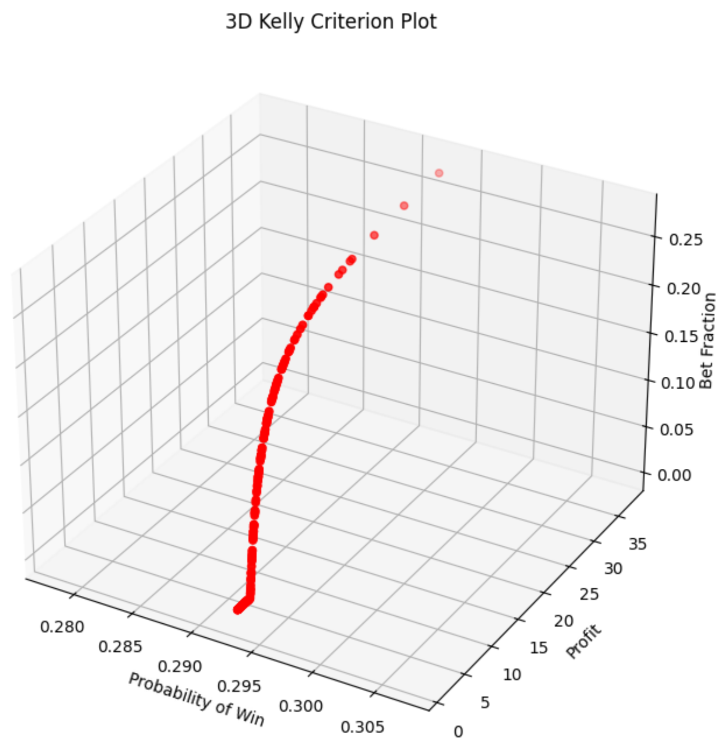


Figure 6.7: The Empirical Kelly Index (Logistic Regression)

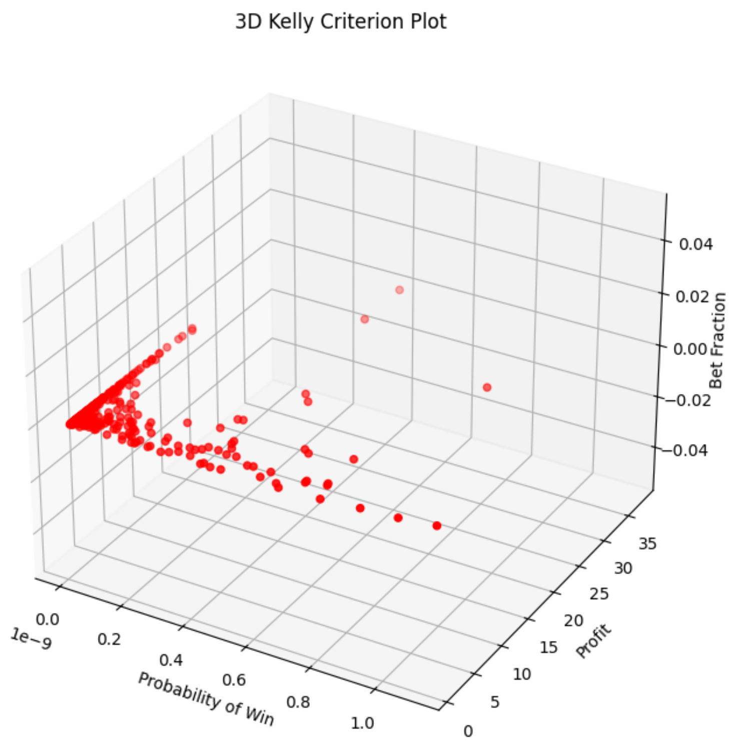


Figure 6.8: The Empirical Kelly Index (Neural Network)

7 Discussion

7.1 Other approaches regarding feature engineering

The data is historical data over time where each match for a team comes after another, which is obvious. This means that the data could be structured as a time series where the next game depends on the last game. The field of time series analysis has yet to be experimented with in this paper but could have huge potential. The data preprocessing was built around seasons and could have been built around a longer time period or a shorter.

As discussed in section 3.2 the transfer window has a strong correlation to a team's performance, the window enable team to sell players that are not preforming well and buy players from other teams that are preforming well. This window is open two times each season once during winter and once during the summer, which could be an argument that the data should be structure for half a season instead of one whole season.

We have a lot of historical data that is consistently structured throughout the dataset. However, football is constantly evolving, and our models do not account for these changes. This issue could be addressed by using fewer data points or by adjusting the data structure based on the era in which the match was played. These hypothetical eras must then be assigned.

Other approaches we could have considered would be to reshape our form-variables to take into account either more or less games. We felt that 5 games was optimal since it is an indication of a team's current performance. However, some teams go on to have great forms over the course of both more or less games, which means that some information might have gone to waste using this approach. In the future, this might be something worth looking into more deeply in order to enhance the predictability as much as possible.

There is no definite way to determine which data prepossessing or feature engineering is the best without testing them all. Therefore we cannot say if any of the previous discussion points would increases the model performance.

7.2 More information that could have been utilized

In football, many factors beyond statistics contribute to a match's outcome. We have previously discussed how the players on a team are a significant factor, but it does not end there. Even if a team has the best player, that player could be injured in a previous game and thus unable to play in the next match. Furthermore, a player does not need to be injured to be unable to play; various other circumstances can affect who gets the chance to play. To utilize this information, our model could benefit from knowing which players are participating in the game. The squad is common knowledge before the game starts, making it possible to incorporate this information into our models.

The managers, who train and instructs the player on what to do also has a big role in the teams performance. Mangers is nothing our models have considered and could ad some valuable information for our models to train on.

Psychology also plays a role in how the match will end, it is hard to monitor or gather data to show this. Some games are more important then others, it could be rivalry or position depending match which contributes to how the player preforms.

Other data that could have been utilized would be for example weather data, does a team perform particularly well under a certain weather condition? Another factor that hugely influences a team's ability to be competitive is the amount of money they are spending. Therefore, it might be interesting to look into the squad's market value, total salary or money spent in a transfer window which might be beneficial for this type of analysis.

7.3 Result reliability

In the betting world there is very small margins that decide if one makes or lose, this was made clear when we tuned our models and very small changes could result in either a profit or a big loss. We only place *bets* over one season and could have done it for multiple seasons and see how much we could make over a longer time period. However, this would lead to a smaller training set.

7.4 Evaluation Metric

For all our models accuracy was the evaluation metric in the training phase to evaluate how well a model preforms. But, that is not what our models were build to predict, our findings were never about correctly predict a match outcome, rather predicting good probabilities to find value bets. A way to get around this is to create your own evaluation metric. In the end trained models were evaluated with the Kelly criterion where we simulated a season and if it would be possible to use this algorithm as an validation metric during training, we might be able to better understand which model performs better than the others.

7.5 Kelly Criterion

The Kelly criterion works best when the probabilities are known and accurate. However, since we are predicting probabilities, we cannot assume they are correct. Our models do not exhibit very high accuracy, and the predicted probabilities vary, indicating that they are not reliable. Calculating probabilities for a football match is inherently challenging due to the game's dynamic nature and the many unforeseen events that can occur. As discussed in section 7.4, using an evaluation metric other than accuracy could improve the effectiveness of the Kelly algorithm. Another approach to optimize the Kelly algorithm is to adjust the dependent variable. In the preprocessing phase of this thesis, it is possible to create three variables corresponding to the different outcomes. Using these three variables, we can develop an algorithm that defines the probabilities for each outcome after the match has been played. With these new variables, we could then perhaps train our models.

7.6 Model building & tuning

Even though we can see some indicators that this is a viable approach and the fact that we have managed to make returns for some of our models on the 2022-2023 premier league season, we are confident that the models can be tuned to be even more precise. As mentioned earlier, the data in it's natural state can be viewed as a multivariate time series, this could potentially be interesting to try and capture when modelling.

For example, the fact that all our models were very biased towards the home side where the probability for this outcome over all of our models were between approximately 32.5 and 100 % indicates that the models aren't capturing a lot of information.

7.6.1 Neural Network

One of the most interesting findings in our opinion were the the results the neural network yielded. Every observation in our bet data got a 100% probability of a home win which was presented in 6.6. This lead to that every bet that was placed was home win and 100% of the bankroll according to the Kelly criterion algorithm. This results in the same fraction as always bet on home win.

With our quite simple architecture the variance in our data was easiest to describe with only which team played at home. As we can see in table 6.1 the neural network did not only generate the highest profit but also almost the highest accuracy.

This finding could be an interesting thing to further investigate. By manipulating the data or changing the architecture to see what happens and figuring out why the neural network always prefers the home team.

As discussed in section 7.3, we work with very small margins, which the network appears to agree with, leading it to often choose the simplest answer. In section 3.3, our data shows that 46% of all the matches in our dataset ended in a win for the home team, which matches the accuracy of the neural network. This suggests that this might be a viable betting strategy in the long run. However, this requires more in-depth research. Conversely, the betting data shows a 48% home win rate, indicating that this particular season had more home wins than average. This suggests that the neural network might not perform as well in other seasons, and the success of this betting strategy in this particular season could be coincidental.

7.6.2 Logistic Regression

Similarly to the neural network, the logistic regression gave a fixed probability for each match. But instead of giving the outcome home win a 100% as the neural network, the logistic regression distributed the probabilities the same way as our *fixed probabilities* in table 3.3. The logistic regression is also a simple model with no tuning that lead to a simple answer. In section 3.2 the home game advantage is presented and now backed up with the results from both the neural network and the logistic regression.

Figures 6.5 and 6.1 describes the distribution for the probabilities set by the logistic regression and the neural network. if these plots are compared to Figure 3.6, it shows more similarities than all of the other models. Not necessarily value vice, more of how the histograms are displayed. So the betting companies probably have a better model to define outliers but our logistic regression and neural network finds the true variance without defining outliers and therefore predicting every probability the same.

Both the logistic regression and the neural network had an element of regularization, the logistic regression used a method called lasso. This regularization technique is the explanation to why all of the variable are the same for each match. lasso sets every variable to zero, except for the home and away team variable. Which also explains why lasso and fixed probabilities in table 3.3 had the same results. In the training data 46% of all matches ended in a home win and was also the probabilities set for a home win by lasso regression.

7.6.3 Rest of the models

XGBoost, Naive Bayess, Support Vector Machine and Random Forest Classifier all had pretty similar results. Regarding of accuracy, Kelly fraction or how the probabilities are distributed. Figures 6.2, 6.3, 6.4 and 6.6 shows that the interval between the highest and lowest probability for each respectively outcome of a match is rather small, which further backs up our finding that when working with bets the margins between profit and loss are very small. The mean probabilities for each outcome very close to to the fixed probabilities indicating that almost every model seams to converges to the outcome distribution presented in table 3.1, expect from the neural network which takes it one step further and sets the home win probability to one.

8 Conclusion

8.1 Question 1

Both yes and no, some of our models have resulted in a profit during this season. But our betting data has small bias with more home win then the rest of the data. However our best model, the neural network yielded a significant profit during the 2022-2023 season. In section 1.2 it is presented that the goal of this thesis was to find the margin between that market influence and the odds set by the bookmakers and having a model that consistently makes money over a long period of time. With the result presented in table 6.1 and the following discussion in section 7 we cant say we have found a way to consistently make money. Even though on paper some of the model have made money their results are not reliable.

With models that highly favor the home team and yields probabilities very close to the outcome distribution presented in Figure 3.1 our results have been based on luck rather than reliable predictions. The bookmakers constantly makes money, it would not be a good business if they did not. So they have figured out something we could not. It is obvious that the bookmakers with their resources would make better models then us, with more experience and computational power

8.2 Question 2

We can't with a 100% certainty conclude that the answer to this question is a pure yes. However, we have seen indications that using data to build statistical models for predictions can be a tool to be utilized among other things when deriving a betting strategy, for the professional gambler. By looking at just pure data, such as we have done over 2022-2023 season, we have seen varied results. Some models have actually made a return while some haven't, and the reliability in our results can't be trusted fully.

A data-driven approach is something that can be used to make more informed decision. But to purely rely on just that and let an algorithm do the job is not a recommendation. As we have previously stated and discussed, there are very small margins you are dealing with when working with betting. In order for this to be a profitable strategy in the long run, over the course of many years and season, one must have domain knowledge. Domain knowledge, in combination with statistical modelling however, might be a recipe for long term success as a player in the betting market.

Bibliography

Azarenka, Ajaxian (2024). *Premier League - All matches from Season 1993-1994 up to date*. https://www.kaggle.com/datasets/ajaxianazarenka/premier-league?resource=download. [Online; DataSet].

Bornn, Luke, Dan Cervone and Javier Fernandez (2018). 'Soccer analytics: Unravelling the complexity of "the beautiful game"'. In: *Significance* 15.3, pp. 26–29.

Casino.org (2016). *A Complete History of Betting Shops in The UK*. https://www.casino.org/blog/a-compete-history-of-betting-shops-in-the-uk/.

Chen, T and C Guestrin (2016). 'XGBoost: A Scalable Tree Boosting System'. In: *CoRR* abs/1603.02754. arXiv: 1603.02754. URL: http://arxiv.org/abs/1603.02754.

Christoffersson, E (2023). *Beating the odds*.

Demmert, Henry G (1973). *The economics of professional team sports*. Lexington, Mass: Lexington Books.

Dijkhuis, Talko B, Matthias Kempe and Koen APM Lemmink (2021). 'Early Prediction of Physical Performance in Elite Soccer Matches—A Machine Learning Approach to Support Substitutions'. In: *Entropy* 23.8, p. 952.

Hastie, T, R Tibshirani and J Friedman (2008). *The Elements of Statistical Learning*. Springer.

James, G, D Witten, T Hastie and R Tibshirani (2020). *An Introduction to Statstical Learning with Applications in R*. Springer.

Krishnan, Gaurav (2023). *Form Score — Understanding The New Football Analytics Metric That Measures A Team's Form*. URL: https://medium.com/after-the-full-time-whistle/form-score-understanding-the-new-football-analytics-metric-that-measures-a-teams-form-99b82b2b113e.

Liashchynskyi, Petro and Pavlo Liashchynskyi (2019). 'Grid search, random search, genetic algorithm: a big comparison for NAS'. In: *arXiv preprint arXiv:1912.06059*.

Lindholm, A, N Wahlström, F Lindsten and T Schön (2022). *Machine Learning A First Course for Engineers and Scientists*. Cambridge University Press.

McCausland, Tammy (2020). *The bad data problem*.

McGrath, Tanner and Brian Pempus (2023). 'How Do Sports Betting Odds Work?' In: *Forbes*. URL: https://www.forbes.com/betting/guide/how-sports-betting-odds-work/.

Noll, Walter, Bernard D Coleman and Walter Noll (1974). 'The thermodynamics of elastic materials with heat conduction and viscosity'. In: *The Foundations of Mechanics and Thermodynamics: Selected Papers*, pp. 145–156.

Poundstone, William (2010). *Fortune's formula: The untold story of the scientific betting system that beat the casinos and Wall Street.* Hill and Wang.

PremierLeague (2024). *Premier League Origins.* `https://www.premierleague.com/history/origins`.

Ramsundar, B and R. B. Zadeh (2016). *TensorFlow for Deep Learning.* O'Reilly.

Ranstam, Jonas and Jonathan A Cook (2018). 'LASSO regression'. In: *Journal of British Surgery* 105.10, pp. 1348–1348.

Rossi, Alessio, Luca Pappalardo, Paolo Cintia, F Marcello Iaia, Javier Fernández and Daniel Medina (2018). 'Effective injury forecasting in soccer with GPS training data and machine learning'. In: *PloS one* 13.7, e0201264.

Turcu, I, GB Burcea, DL Diaconescu, MCR Barbu, MC Popescu and P Apostu (2020). 'The impact of the betting industry on sports'. In: *Bulletin of the Transilvania University of Braşov. Series IX: Sciences of Human Kinetics*, pp. 251–258.

Wikipedia contributors (2024a). *Home advantage — Wikipedia, The Free Encyclopedia.* `https://en.wikipedia.org/w/index.php?title=Home_advantage&oldid=1193588812`. [Online; accessed 29-March-2024].

– (2024b). *Kelly criterion — Wikipedia, The Free Encyclopedia.* `https://en.wikipedia.org/w/index.php?title=Kelly_criterion&oldid=1214651753`. [Online; accessed 28-March-2024].

– (2024c). *Premier League — Wikipedia, The Free Encyclopedia.* `https://en.wikipedia.org/w/index.php?title=Premier_League&oldid=1216030089`. [Online; accessed 28-March-2024].

Appendix A

Acknowledgments

During the preparation of this work the authors have used ChatGPT 4 in order to debugg code that did not work. After using this tool, the authors reviewed and edited the content as needed and takes full responsibility for the content of the publication.

To get the code please contact one of the authors.

Oscar Båth - Oscarbw99@gmail.com

Viktor Sjöberg - viktor@alfrida.se

Appendix B

Graphs

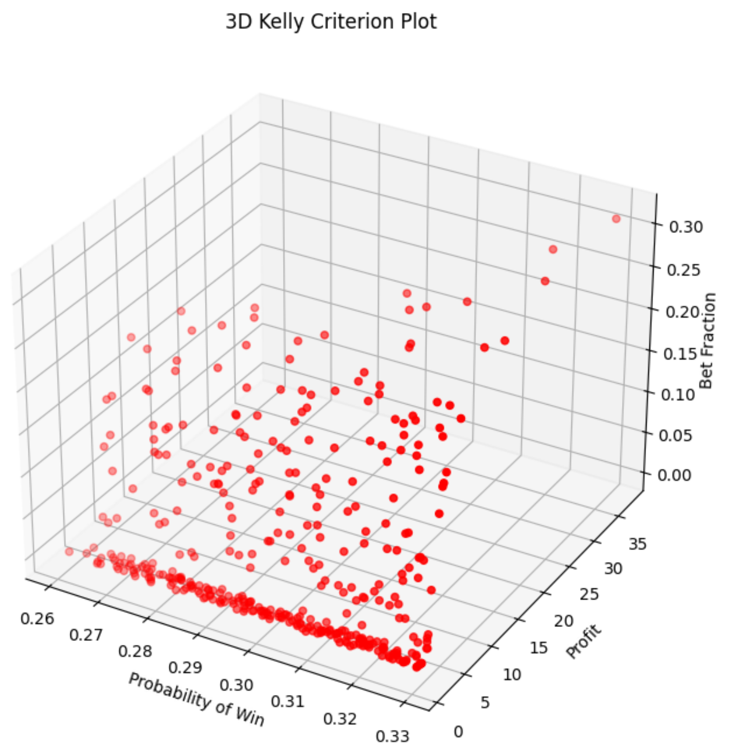


Figure B.1: The Empirical Kelly Index (Navie Bayes)

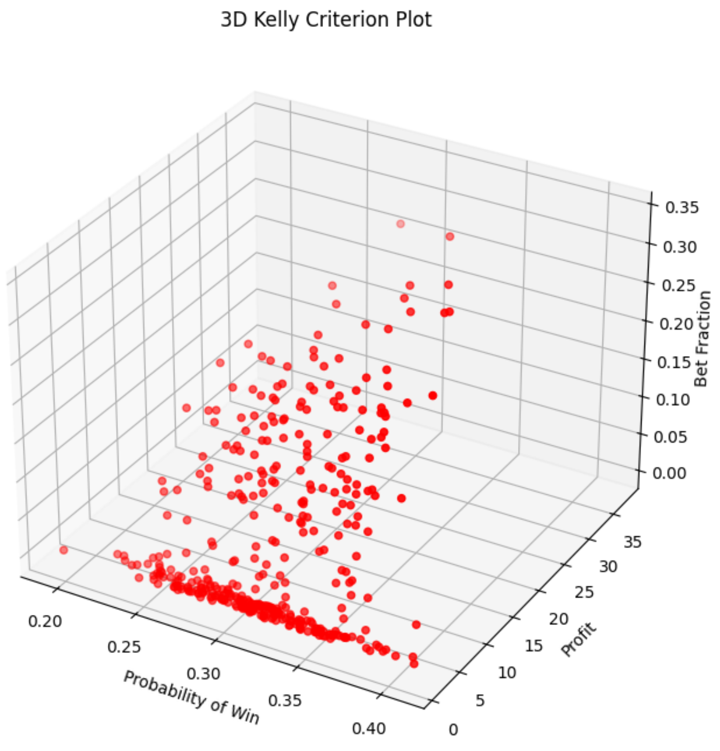


Figure B.2: The Empirical Kelly Index (Random Forrest Classifier)

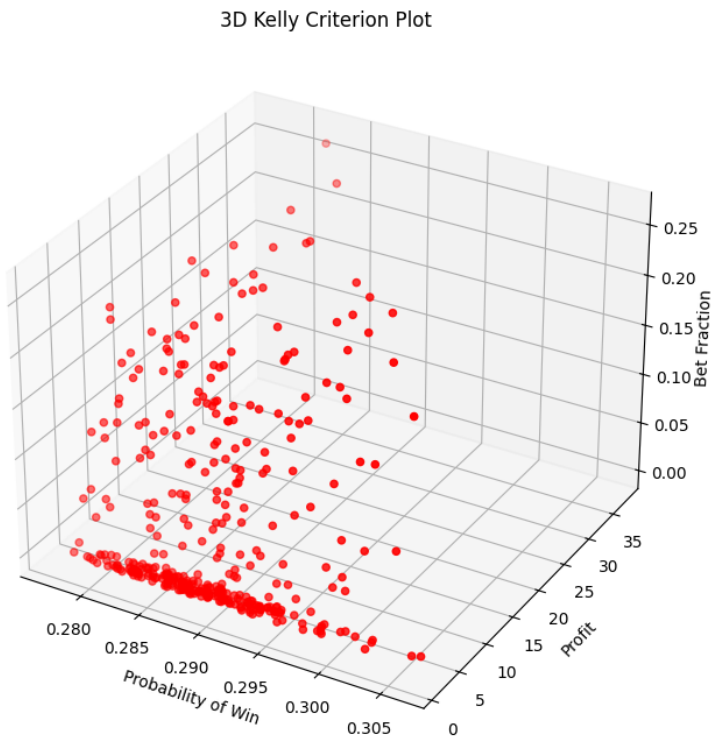


Figure B.3: The Empirical Kelly Index (Support Vector Machine)

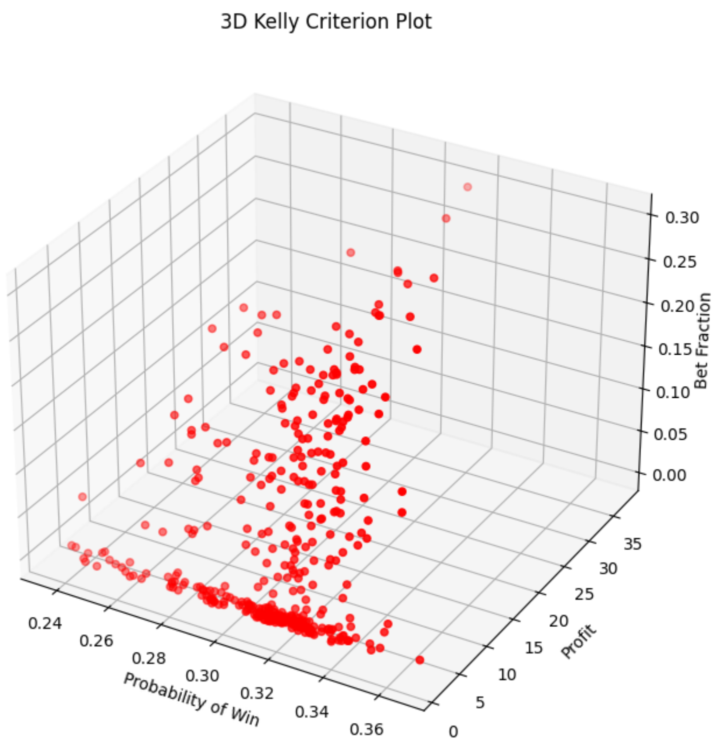


Figure B.4: The Empirical Kelly Index (XGBoost)