

Eirik S. Nielsen & Sander F. Sandøy

Profiting from Football Betting using Artificial Neural Networks

Master's thesis in Computer Science

Supervisor: Helge Langseth

June 2019

Eirik S. Nielsen & Sander F. Sandøy

Profiting from Football Betting using Artificial Neural Networks

Master's thesis in Computer Science
Supervisor: Helge Langseth
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

 **NTNU**
Norwegian University of
Science and Technology

Abstract

The football betting market is an enormous market, with numerous bookmakers allowing hopeful punters to place bets on almost every aspect of a match. The most popular aspect of football betting is placing bets on the final match outcome. Predicting these outcomes is a difficult task given the complexity of the game, but by consistently producing better predictions than the bookmaker, an opportunity to make a profit on the market may arise.

In this thesis, two match prediction models using artificial neural networks are developed, where one is a fully data-driven adaptation of a successful Bayesian network model utilizing domain expert knowledge. The effect of utilizing domain knowledge in the structure of neural networks is explored by grouping related input features into separate sub-networks.

The profitability of the models is evaluated over two seasons of the English Premier League using different money management strategies. Experiments in using reinforcement learning methods to train a money management agent are performed to explore the possibility of building a complete end-to-end betting system.

Some model-strategy combinations were able to generate a profit over both test seasons, showing that it is indeed possible to profit from football betting using artificial neural networks.

Sammendrag

Markedet for fotballtipping er enormt, og mange tippeselskaper lar håpefulle spillere plassere veddemål på omtrent alle aspekter av en kamp. Det mest populære veddemålet er å spille på kampens sluttresultat. Å forutsi disse utfallene er en vanskelig oppgave på grunn av spillets kompleksitet, men ved å konsekvent produsere bedre forutsigelser enn tippeselskapene kan muligheten for profitt vise seg.

I denne oppgaven blir to kamppredikeringsmodeller basert på nevrale nettverk utviklet, hvor en er en fullstendig datadreven tilpasning av en vellykket Bayesiansk nettverks-modell avhengig av kunnskapen til en domeneekspert. Effekten av å utnytte domenekunnskap i strukturen av nevrale nettverk er utforsket ved å gruppere relaterte inputvariable i separate sub-nettverk.

Lønnsomheten til modellene er evaluert over to sesonger av den engelske Premier League med ulike spillstrategier. Eksperimenter med å bruke metoder fra reinforcement learning for å trene en spillstrategi-agent blir utført for å utforske muligheten til å bygge et komplett ende-til-ende spillsystem.

Enkelte modell-strategi-kombinasjoner klarte å generere profitt over begge test-sesongene, som viser at det er mulig å profitere fra fotballtipping med kunstige nevrale nettverk.

Preface

This master thesis has been carried out at the Department of Computer Science at the Norwegian University of Science and Technology from January to June 2019.

We would like to thank our supervisor, Helge Langseth, for his guidance and honest feedback throughout this work. Thanks also to our classmates and friends, the BlockBoys, for all the lunches and coffee breaks. Last but not least, thanks to Marthe Linnea Tronrud and Ine Kristine Lading for keeping us somewhat sane through the ups and downs of the past year.

Eirik S. Nielsen & Sander F. Sandøy
Trondheim, June 1, 2019

Contents

1	Introduction	1
1.1	Goals and Research Questions	2
1.2	Thesis Structure	3
2	Background Theory	5
2.1	Artificial Neural Networks	5
2.2	Reinforcement learning	8
2.2.1	Markov Decision Processes	9
2.3	Measuring the accuracy of a prediction model	10
2.4	Odds features	14
2.4.1	Margin	15
2.4.2	Bias	16
2.5	Expected goals	17
3	Related Work	19
3.1	Previous match outcome prediction models	19
3.1.1	pi-football	19
3.1.2	FiveThirtyEight’s Club Soccer Prediction	22
3.1.3	Beating the Bookmakers	24
3.2	Money Management	26
4	Models	31
4.1	Data-driven pi-football	31
4.2	Expected goals of starting lineup	33
4.3	Network Structure	36
4.3.1	Model component grouping	38
4.4	Money Management	39
4.4.1	Weekly Betting Environment	40

4.4.2	Perfect Information Environment	42
4.4.3	Single Match Environment	46
5	Data and System Design	51
5.1	Data sources	51
5.1.1	Understat	51
5.1.2	FiveThirtyEight	54
5.1.3	Football-Data	55
5.2	Data collection	55
5.2.1	Understat scraper	55
5.2.2	Football-Data scraper	56
5.2.3	FiveThirtyEight	56
5.3	System Design	56
5.3.1	Database	56
5.3.2	Neural networks	57
5.3.3	Betting simulator	57
6	Experiments and Results	59
6.1	Experimental Plan	59
6.2	Experimental Setup	60
6.2.1	Network setup	60
6.2.2	Data	61
6.2.3	Betting evaluation	63
6.3	Experimental Results	64
6.3.1	Baselines	64
6.3.2	Fully connected expected goals of starting lineup	65
6.3.3	Expected goals of starting lineup with sub-networks	71
6.3.4	Fully connected pi-football	76
6.3.5	Data-driven pi-football with sub-networks	81
7	Discussion	87
7.1	Model performance comparison	87
7.2	Betting strategy performance	89
7.3	Effect of domain knowledge structuring	90
7.4	Domain knowledge limitations	91
7.5	Model selection limitations	91
7.6	Money management agent	92
8	Conclusion and Future Work	95
8.1	Conclusion	95
8.2	Future Work	96

8.2.1	Threshold for expected gain	96
8.2.2	Betting-based validation criteria	97
8.2.3	Considering multiple bookmakers	97
8.2.4	More complex action space for betting agent	97
A	Database tables overview	99
	Bibliography	101

List of Figures

2.1	Perceptron	6
2.2	Neural Network	7
2.3	The agent-environment interaction in an MDP, from Sutton and Barto [2018]	9
2.4	Predicted probabilities by hypothetical prediction models	11
2.5	Applying the specified scoring rules to each benchmark presented in Figure 2.4. Taken from Constantinou and Fenton [2012]	13
2.6	Scores generated by the RPS for each hypothetical model. Taken from Constantinou and Fenton [2012]	14
2.7	Scoring zone	18
3.1	Overview of pi-football, from Constantinou et al. [2012]	20
3.2	Simplified topology of revised pi-football, from Constantinou et al. [2013]	22
3.3	SPI-score	24
4.1	Neural-network-architecture	37
4.2	Accumulated reward per episode, Weekly Environment	41
4.3	Accumulated reward per episode, Perfect Information Environment, Feasibility Reward	43
4.4	Distribution of bets made by agent trained in Perfect Information Environment using Feasibility Reward	44
4.5	Accumulated reward per episode, Perfect Information Environment, Profit Reward	45
4.6	Distribution of bets made by agent trained in Perfect Information Environment using Profit Reward	46
4.7	Accumulated reward per episode, Real Data Environment, Profit Reward	48

4.8	Aggregated distribution of bets made by agent trained in Real Data Environment	49
5.1	Overview of Chelsea's player-statistics during the 2016/2017-season and the expected values for each player, taken from Understat.com. The green and red numbers denotes the difference between expected goals and actual goals.	53
5.2	Overview of the database used by the system.	57
6.1	Predicted probabilities compared to actual outcomes, fully connected expected goals of starting lineup.	67
6.2	ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using the fully connected expected goals of starting lineup model.	68
6.3	ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using the fully connected expected goals of starting lineup model.	69
6.4	Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.	70
6.5	Predicted probabilities compared to actual outcomes, expected goals of starting lineup with sub-networks.	72
6.6	ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using the expected goals of starting lineup model with sub-networks.	73
6.7	ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using the expected goals of starting lineup model with sub-network.	74
6.8	Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.	75
6.9	Predicted probabilities compared to actual outcomes, fully connected pi-football.	77
6.10	ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using fully connected data-driven pi-football model.	78
6.11	ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using fully connected data-driven pi-football model.	78

6.12	Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.	80
6.13	Predicted probabilities compared to actual outcomes, pi-football with sub-networks.	82
6.14	ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using data-driven pi-football model with sub-networks.	83
6.15	ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using data-driven pi-football model with sub-networks.	83
6.16	Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.	85
7.1	FiveThirtyEight's predicted probabilities compared to actual outcomes	93

List of Tables

5.1	Description of match level data from Understat.com	52
5.2	Description of player level data from Understat.com	52
5.3	Overview of the data available from Soccer Power Index.	54
6.1	Return on investment for three benchmarks using the fixed bet strategy	65
6.2	Calculated margin for William Hill in the English Premier League .	65
6.3	Fully connected expected goals of starting lineup - validation results	66
6.4	ROI after the 2016/2017 and 2017/2018 seasons for all betting strategies for the fully connected expected goals of starting lineup-model.	69
6.5	Expected goals of starting lineup with sub-networks - validation results	71
6.6	ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the expected goals of starting lineup-model with sub-networks.	74
6.7	Fully connected pi-football - validation results	76
6.8	ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the fully connected pi-football-model.	79
6.9	Pi-football with sub-networks - validation results	81
6.10	ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the pi-football-model with sub-networks. . . .	84
7.1	Mean ROI for the models after the 2016/2017 season	88
7.2	Mean ROI for the models after the 2017/2018 season	88

Chapter 1

Introduction

Association Football (hereafter referred to as simply *football*) is a sport which engages fans all over the world. In a professional game played in a top league in Europe, several thousands of fans can be present at the stadium and millions could be watching on the television.

Due to its enormous popularity and interest, football has also become very popular in the field of sports betting. Many people dedicate all of their spare time to try and analyze every inch of this game so that they may be able to make accurate predictions of multiple aspects in a game of football, whether it is predicting the final result, how many goals are scored in the game or which player receives a red card. Being able to correctly predict these outcomes is not so straightforward, however. There are an enormous number of factors that may contribute to the final result in a game of football. Which players are playing, the strengths of each team, who are playing at home and recent form are all examples of factors that could influence the outcome of a game.

Even with so many different factors and the high complexity of the game, several researchers have taken an interest in building models which can reliably predict the outcomes of matches. These models have mostly been based on statistical data, and they have yet to provide bettors a major breakthrough which can make them able to beat the bookmakers at their own game. In the current age of Big Data, sports are no exception; services like Understat.com and Opta Sports gather and store data from thousands of games in all kinds of sports worldwide. This data foundation, combined with enormous quantities of historical data which these companies also supply, may enable new angles and interesting models, if one

is able to manage it correctly.

The focus and motivation for this thesis is to create models for predicting the outcomes of football matches and generating a profit on the betting market. By combining the force of machine learning methods and the vast data foundation we aim to create a data-driven system able to do such. This report will explore how different machine learning methods, specifically artificial neural networks and reinforcement learning, can be used for this purpose.

1.1 Goals and Research Questions

To guide our research throughout this project, the following research goal has been formulated:

Goal *Explore the possibilities of generating a profit on the football betting market using data-driven machine learning methods.*

In order to achieve this goal, the following research questions were formed:

Research question 1 *Can football matches be adequately predicted by artificial neural networks?*

Betting companies provide odds on match outcomes for football matches, and these can be viewed as a prediction of the outcome by the bookmakers. For a prediction model to be successful, it needs to model the true probability distribution for match outcomes for matches better than what the bookmakers do. Our thesis aims to find out if the power of artificial neural networks and its ability to extract patterns from a large set of features can be applied to adequately predict the outcome of football matches.

Research question 2 *Does utilizing domain knowledge in the structure of artificial neural networks increase their predictive performance?*

In a fully connected feed-forward neural network trained by gradient descent back-propagation, the connections between neurons are strengthened or weakened during training to try to capture their related effect on the output. As the designers

of an experiment, we may already have a preconceived notion that certain input features might be more closely related than others. By using this domain knowledge, the network could be constructed in such a way that related features are grouped together and isolated from the rest of the features for a number of layers, allowing the network to find valuable intermediary representations, before treating the problem as a whole. This research question concerns the performance of such a network, compared to a network using the same features, but in a more traditional fully connected setup.

Research question 3 *Under what assumptions can standard reinforcement learning methods be used to learn an approximately optimal money management strategy?*

When using artificial neural networks as a predictor, it is easy to think of the output of a softmax layer as representing the model's confidence. However, that is not the case, as a model can be uncertain in its predictions even with a high softmax output (Gal [2016]). As most money management strategies for sports betting use the punter's perceived probability of each outcome to determine how to place bets, using the output of a prediction model might not yield the best results. This output might not directly represent the model's confidence in each outcome.

Because of this, we wish to explore the possibilities of using reinforcement learning methods to train a money management system, using the output of the prediction models as input, yielding a complete end-to-end betting system.

1.2 Thesis Structure

This section presents an overview of the rest of the thesis.

- **Chapter 2** presents relevant background theory in order to understand the contents of this report, such as artificial neural networks, reinforcement learning and an accuracy metric suitable for outcome prediction models.
- **Chapter 3** presents related research on football match prediction and money management strategies.

- **Chapter 4** presents the proposed match prediction models, as well as exploratory experiments in using reinforcement learning for money management.
- **Chapter 5** presents the data sources used for the experiments conducted in this project, as well as a brief overview of the software components built to facilitate these experiments.
- **Chapter 6** presents the experiments conducted and their results in terms of both accuracy and profitability.
- **Chapter 7** presents a discussion of the performance and limitations of the different prediction models and money management strategies.
- **Chapter 8** presents the conclusion and summarizes our findings in light of the research questions. Suggestions for future improvement of the system and experiments are also presented.

Background Theory

This chapter presents the background theory which is essential for understanding the upcoming chapters in this report. Section 2.5 and Section 2.1 were part of a specialization project done at NTNU (Nielsen and Sandøy [2018]).

2.1 Artificial Neural Networks

Artificial neural networks (ANN) are a computational model inspired by neuroscience. The concept enables a computer to learn from observational data and it can be used to extract patterns and recognize trends in the data that are too complex for a human to process. In particular, the model is based on the hypothesis that human mental activity consists primarily of electrochemical activity in networks of brain cells called **neurons** (Russell and Norvig [2016]). Inspired by this hypothesis, some of the earliest AI work aimed to create **artificial neural networks**, and McCulloch and Pitts [1943] devised a simple mathematical model of a human neuron. This concept was extended by Rosenblatt [1958] and he called this model a *perceptron*. This is illustrated in Figure 2.1. The concept is that a perceptron *activates* when a linear combination of its inputs exceeds some threshold, much like how human neurons activates when given a certain stimulation.

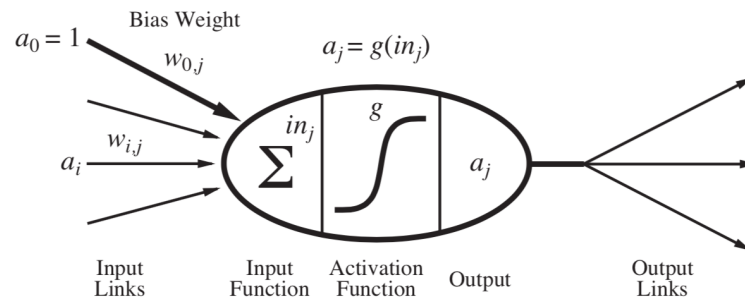


Figure 2.1: A simple mathematical model for a neuron. Taken from Russell and Norvig [2016]

By chaining perceptrons together in a layered network and by varying the weights and activation thresholds, one can get different models of decision-making, which makes neural networks a very flexible architecture. Perceptrons in each layer can make decisions by weighing up the results from the preceding layer which makes the decision more complex and abstract. Multilayer networks can engage in sophisticated decision making. A neural network is illustrated in Figure 2.2.

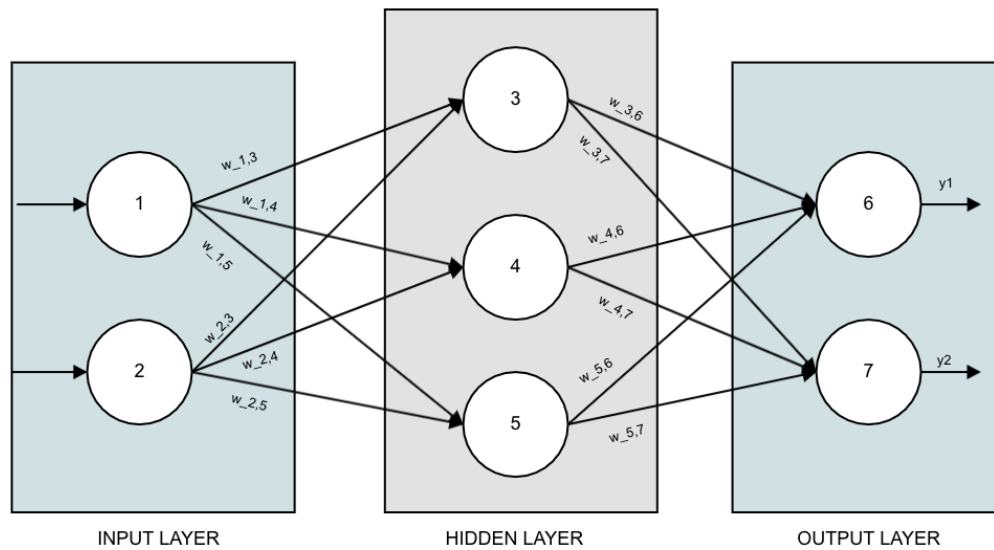


Figure 2.2: A simple feed-forward neural network with one hidden layer. Here one can see the directed links between the nodes in each layer and its corresponding weights.

The weights between layers in a neural network are tuned by a process called **backpropagation**. After the network has forward-propagated its input signals and calculated an output, the backpropagation phase begins. The idea of backpropagation is to tweak the generated output toward the desired target output. Here the generated output is compared to the target output using a *loss function*. This loss function calculates the difference between the generated output and the target output. Since the input is constant for each case, it is only the weights that need to be tweaked, and the idea is that changes in the weights cause changes in the output. After calculating the error, the errors propagate backwards through the network, calculating an error gradient value for each neuron in the layers, with respect to the weights.

These errors are then fed to a *optimization function*, which adjusts the neuron weights with respect to minimizing the loss calculated by the loss function. *Gradient Descent* is a very common optimization function which is used in backpropagation. This algorithm follows the *gradient* of the function to be optimized until a minimum is reached. The speed of the convergence is controlled by the *learning rate* α , which needs to be chosen carefully. Too high learning rate can lead to the algorithm becoming unstable, while too low learning rate can lead to getting stuck in local minima. Algorithm 1 displays the Gradient Descent Algorithm.

Algorithm 1 Gradient Descent algorithm

```

w ← tensor of all weights in the network
α ← learning rate
loop until convergence do
  for each  $w_i$  in w do
     $w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \text{Loss}(\mathbf{w})$ 

```

This gives neural networks an ability to generalize and to respond to unexpected inputs in pattern recognition. The neurons are taught to recognize various specific patterns and whether or not to fire when that pattern is given as input. If given enough examples, a network can then *generalize* and learn the relationships between the input variables. In our case, a neural network may be able to spot the relationships between the features we think have the most impact on a match result and also recognize some patterns that are unknown to us when predicting a match outcome.

2.2 Reinforcement learning

Reinforcement learning is a class of machine learning methods where an *agent* learns to achieve a *goal* by performing *actions* on its *environment*. Unlike supervised learning methods, there is no training set of labeled examples. Instead, the agent must explore its environment, and observe how it changes based on its actions. The following explanation of reinforcement learning agents is adapted from Sutton and Barto [2018].

Beyond the agent itself and the environment it operates in, there are four main elements of a reinforcement learning system: a *policy*, a *reward signal*, a *value function*, and optionally a *model* of the environment (Sutton and Barto [2018]).

The *policy* π_t defines the agent's behavior at timestep t , and is a function mapping the observed *state* s_t of the environment with state space \mathcal{S} , to an *action* a_t in the agent's action space \mathcal{A} , i.e

$$\pi : \mathcal{S} \rightarrow \mathcal{A} \tag{2.1}$$

The *reward signal* is sent from the environment to the agent on each timestep, and consists of a single number called the *reward*. The reward signal defines the goal of the agent, as the agent's objective is to maximize its total accumulated reward. This reward signal is used to alter the agent's policy; if the selected action results

in a low reward, the policy may be changed to prefer another action in the given situation in the future.

Where the rewards signalize the immediate desirability of an environmental state, the *value function* evaluates how beneficial a given state is in the long term. The *value* of a state represents the total amount of reward the agent can expect in the future, starting from this state using its current policy π . When choosing between possible actions, the agent bases its choice on the values of the resulting states, and not the rewards. The problem lies in the fact that the values are harder to determine than the rewards, generally requiring estimations and recalculation as the agent explores more of the environment. As this value estimation governs the actions made by the agent, a method for estimating the value of states is the most important component of most reinforcement learning algorithms.

The final element used in some reinforcement learning systems is a *model* of the environment. The model is the agent's internal representation of the environment's behavior, and allows the agent to make inferences of how the environment will change based on its actions. *Model-based* reinforcement learning methods generally use their model for *planning*, while *model-free* methods take a simpler trial-and-error approach to learning.

2.2.1 Markov Decision Processes

The *Markov decision process* (MDP) is the mathematical foundation for most reinforcement learning systems, and frames the problem of learning from interaction to achieve a goal. Central to this process is the interaction between the *agent* and the *environment*, where the agent selects and performs an action, and the environment responds.

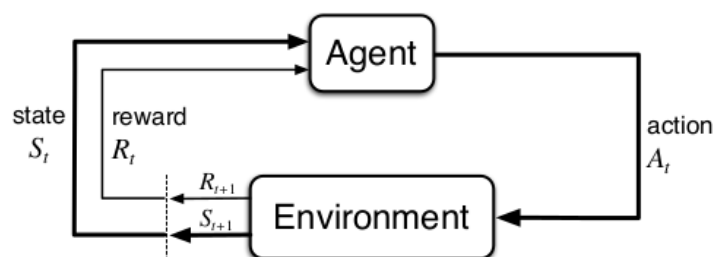


Figure 2.3: The agent-environment interaction in an MDP, from Sutton and Barto [2018]

The interaction is formalized over a sequence of discrete time steps, $t = 0, 1, 2, \dots$. At each time step t , the agent receives a representation of the environment's *state*, $S_t \in \mathcal{S}$, and based on this state, the agent selects an *action*, $a_t \in \mathcal{A}(s)$. On the subsequent time step, the agent then receives a *reward* $r_{t+1} \in \mathcal{R} \subset \mathbb{R}$, as well as the resulting next state S_{t+1} .

When the state, action and reward spaces $(\mathcal{S}, \mathcal{A}, \mathcal{R})$ have a finite number of elements, the *dynamics* of the MDP can be defined by

$$p(s', r | s, a) = \Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\} \quad (2.2)$$

for all $s', s \in \mathcal{S}$, $r \in \mathcal{R}$ and $a \in \mathcal{A}$.

The defining characteristic of an MDP is that the probability of each possible value for S_t and R_t is dependent *only* on the immediately preceding state and action S_{t-1} and A_{t-1} , not on a sequence of preceding states and actions. This is called the *Markov property*, and should be seen as a property of the *state*, more than a restriction on the actions.

Because of the Markov property, Equation 2.2.1 completely characterize the environment, and interesting properties of the environment can be computed from it. For instance, the *state-transition probabilities*

$$p(s' | s, a) = \Pr\{S_t = s' | S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in \mathcal{R}} p(s', r | s, a)$$

and the expected reward for a given state-action pair

$$r(s, a) = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r | s, a)$$

The Markov Decision Process serves as an abstraction of goal-directed learning from interaction, reducing the process down to three signals being passed between the learning agent and the environment. Despite its simplicity, it has nevertheless shown to be useful and applicable to a lot of problems. Much of the challenge in using this abstraction is in the representation of the states and actions, as well as actually formulating the given problem as a formal MDP.

2.3 Measuring the accuracy of a prediction model

Constantinou and Fenton [2012] outline in their paper that measuring the accuracy

of a prediction model is a crucial part of its validation. When determining the accuracy of prediction models so-called *scoring-rules* are used and these assign a numerical score to each prediction based on how "close" the probabilities are to the actual observed outcome. Defining suitable scoring rules for match outcome forecasts in football matches has proven to be extremely difficult (Constantinou and Fenton [2012]). The authors observe that the set of outcomes (home, away, draw) must be considered on a *ranked* scale, which means that the outcomes are ranked by how close they are together. A draw outcome for a match is closer to an away outcome than an away outcome is to a home outcome, since only one goal is required to go from a draw to an away outcome.

Two hypothetical prediction models, α and β , are presented in their paper and Constantinou and Fenton [2012] evaluate these models by looking at their predicted probability distribution compared to the actual match outcome. Figure 2.4 shows the two predictions for the two models for each of the five matches.

Match	Model	$p(H)$	$p(D)$	$p(A)$	Result	'Best model'
1	α	1	0	0	H	α
	β	0.9	0.1	0		
2	α	0.8	0.1	0.1	H	α
	β	0.5	0.25	0.25		
3	α	0.35	0.3	0.35	D	α
	β	0.6	0.3	0.1		
4	α	0.6	0.3	0.1	H	α
	β	0.6	0.1	0.3		
5	α	0.5	0.45	0.05	H	α
	β	0.55	0.10	0.35		

Figure 2.4: Predicted probabilities by the two hypothetical prediction models, α and β , for five hypothetical matches. Taken from Constantinou and Fenton [2012]

As can be seen from Figure 2.4 model α is the model which scores the higher in each of the matches and Constantinou and Fenton [2012] argues that this is because:

- **Match 1:** Model α predicts the actual outcome with total certainty and hence must score higher than β .
- **Match 2:** Both models assign the highest probability to the winning outcome H, with the remaining two outcomes evenly distributed. Since the

observed value of α is higher than that of β , it must score higher.

- **Match 3:** Both models assign the same probability to the correct outcome, but model α is more accurate since its overall distribution is more indicative of a draw.
- **Match 4:** Both models assign the same probability to the correct outcome, but model α is more accurate since its overall distribution is more indicative of a home victory.
- **Match 5:** In this example β predicts the correct outcome with a higher probability than α . However, the distribution of model α is more indicative of a home win than model β . Constantinou and Fenton [2012] explains this by considering a gambler who is confident that the home team will not lose, and seeks to place a lay bet (a bet that is successful if the home team wins, or the match ends with a draw). If α and β are forecasts by two different bookmakers, bookmaker α will pay less for the winning bet.

They then present five different scoring rules, and show why these are not able to correctly evaluate the accuracy of two hypothetical prediction models. Figure 2.5 shows how these scoring rules score models α and β . A tick means that the scoring rule correctly scores model α higher than β . A double cross means that the scoring rule scores β higher than α , and a single cross means that the scoring rule returns the same value for both models.

Match (Model)	Binary Decision Score	Brier Score	Geometric Mean Score	Information Loss Score	MLE Score
1	✓	✓	✓	✓	✓
(α)	1	0	1	0	0
(β)	0	0.02	0.9	0.152	-0.1054
2	✗	✓	✓	✓	✓
(α)	1	0.06	0.8	0.3219	-0.2231
(β)	1	0.375	0.5	1	-0.6931
3	✗	✓	✗	✗	✗
(α)	0	0.735	0.3	1.7369	-1.2039
(β)	0	0.86	0.3	1.7369	-1.2039
4	✗	✗	✗	✗	✗
(α)	1	0.26	0.6	0.7369	-0.5108
(β)	1	0.26	0.6	0.7369	-0.5108
5	✗	✗ ✗	✗ ✗	✗ ✗	✗ ✗
(α)	1	0.455	0.5	1	-0.6931
(β)	1	0.335	0.55	0.8625	-0.5978

Figure 2.5: Applying the specified scoring rules to each benchmark presented in Figure 2.4. Taken from Constantinou and Fenton [2012]

As can be seen from the Figure, none of the scoring rules returns the "correct" outcome for all 5 scenarios. Indeed, all of the scoring rules fail to correctly identify model α as superior for scenarios 4 and 5.

Subsequently, Constantinou and Fenton [2012] present the Rank Probability Score (RPS) as an alternative scoring. RPS was introduced by Epstein [1969] and has been described as a particularly appropriate scoring rule for evaluating probability forecasts of ordered variables. Equation 2.3 presents the RPS

$$RPS = \frac{1}{r-1} \sum_{i=0}^{r-1} \left(\sum_{j=0}^i (p_j - e_j) \right)^2 \quad (2.3)$$

where r is the number of outcomes ($r=3$ for football matches), p_j is the predicted probability for outcome j and e_j is the actual observed outcome for j . Figure 2.6 presents the generated score for each of the scenarios presented in Figure 2.4, along with the respective cumulative distributions. A lower score indicates a better forecast, and one can see that the RPS, unlike the previous metrics, manages to

correctly score α as the best model for all 5 matches. Constantinou and Fenton [2012] suggest using either the arithmetic mean over the individual scores, or the total of the individual scores when using RPS to evaluate a prediction model over several matches.

Match	Model	$\sum_{j=1}^{i=1,2,3} p_j$	$\sum_{j=1}^{i=1,2,3} e_j$	RPS
1	α	1,1,1	1,1,1	(0.0000)
	β	0.90,1,1	1,1,1	0.0050
2	α	0.80,0.90,1	1,1,1	(0.0250)
	β	0.50,0.75,1	1,1,1	0.1562
3	α	0.35,0.65,1	0,1,1	(0.1225)
	β	0.60,0.90,1	0,1,1	0.1850
4	α	0.60,0.90,1	1,1,1	(0.0850)
	β	0.60,0.70,1	1,1,1	0.1250
5	α	0.50,0.95,1	1,1,1	(0.1262)
	β	0.55,0.65,1	1,1,1	0.1625

Figure 2.6: Scores generated by the RPS for each hypothetical model. Taken from Constantinou and Fenton [2012]

2.4 Odds features

In order to understand how to make a profit in sports betting, it is useful to first examine how the odds for match outcomes are formed.

Bookmakers who offer odds on match outcomes are interested in making a profit. They want to ensure that no matter the final outcome of a match, they are still left with a profit. There are several features of sports betting odds which can impact both the betting agent's selection of bets to take and the profit for the bookmakers. The bookmakers try to take advantage of these features and in this section we will highlight some of them.

2.4.1 Margin

The added *margin* to each match is one way for the bookmakers to ensure profit. Vlastakis et al. [2009] illustrate that the expected margin for an event with n possible outcomes can be represented as

$$E(M) = 1 - \sum_{i=1}^n P_i * \omega_i * d_i \quad (2.4)$$

where i corresponds to a match outcome (home, draw or away victory) and $n = 3$ for football matches. According to Equation 2.4, the expected margin (M) on each match depends on the probability associated with each outcome (P_i), the percentage of bets placed on each outcome (ω_i) and the given odds (d_i) for each outcome. This equation implies that there are several ways in which the bookmakers can set their prices to achieve the highest amount of profit. For example, bookmakers may try to forecast game outcomes as accurately as possible so that the odds reflect these expectations. Alternatively, they may try to forecast the distribution of bets on each outcome. A third option is that the bookmakers use a combination of these two approaches.

Since the true probability of each outcome is not known the bookmakers can only control the values of d_i . The values of ω_i often changes rapidly (often corresponding to the values of d_i), and are controlled by the bettors. The bookmakers must thus ensure that

$$\forall i \in 1, 2, 3 : \omega_i * d_i < 1 \quad (2.5)$$

If $\omega_i * d_i > 1$ and i is the actual outcome of the match then the bookmakers will receive a negative profit for that given match.

To calculate the actual margin that bookmakers earn from a single match Equation 2.4 displays the need to know both the odds on each outcome as well as the distribution of bets across these outcomes. The odds are publicly available, but this is not the case for the bet-distributions. This means that one cannot calculate the actual margin for bookmakers, but rather an *implied* margin, denoted here as M' . This is estimated by assuming that the bets are equally distributed across outcomes and that the odds are set according to the true probabilities (Vlastakis

et al. [2009]). That is,

$$\forall i \in 1, 2, 3 : \omega_i = \frac{1}{3} \quad (2.6)$$

It is assumed that the odds are set on the basis of the true probabilities of each outcome. The fair odds on an outcome is simply the reciprocal of the probability P_i of the occurrence of that outcome. However, if odds were priced exactly at their fair level according to the true probabilities then the expected bookmaker gain would be zero (this can be easily seen if one replaces d_i with P_i in Equation 2.4). For this reason, actual odds are somewhat smaller than fair odds in order to allow a positive margin for bookmakers (Vlastakis et al. [2009]). Accordingly, actual odds do not correspond to true probabilities but to somewhat larger *implied* probabilities (denoted P'_i). The expected implied margin can be estimated as

$$E(M') = \left(\sum_{i=1}^n P'_i \right) - 1 = \left(\sum_{i=1}^n \frac{1}{d_i} \right) - 1 \quad (2.7)$$

When Liverpool FC played at home against Burnley FC on March 10, 2018 in the English Premier League the average odds were 1.17, 7.96, 18.46 for each of the match outcomes (home, draw and away, respectively). Inserting these odds into Equation 2.7 the implied margin for the bookmakers was $\frac{1}{1.17} + \frac{1}{7.96} + \frac{1}{18.46} - 1 = 0.034$. This means that the bookmakers were expected to gain a profit of 3.4% for this given match. Thus, in order for football prediction models to make a profit the models must also beat this built-in margin.

2.4.2 Bias

Another mechanism in the gambling markets in the bookmakers' advantage are *odds biases*. These biases are biases which have been observed for different kinds of bets, and they are important to consider when placing bets, as they can strongly impact the profitability of a strategy. Constantinou and Fenton [2013] presents three odds biases which can be utilized when creating a betting strategy for prediction models:

- **Favorite-longshot bias:** This bias implies that bets placed on favourites yield a higher return than bets placed on longshots. According to Constantinou and Fenton [2013], this is caused by the bettor's preference for backing

risky outcomes (longshots). This means that bettors prefer betting on outcomes which give a higher return rather than betting on a “safe” outcome which would need the bettor to stake more money on the given outcome. Bookmakers are believed to exploit these types of behaviour and increase profitability by offering more-than-fair odds for “safe” outcomes, and less-than-fair odds for “risky” outcomes.

- **Home-away bias:** Constantinou and Fenton [2013] demonstrates that betting on home wins yields the highest cumulative return under most of the experimented scenarios. This phenomenon does not appear to be particularly profitable, but it shows that the home-away bias appears to be equally as strong as the favorite-longshot bias. Here, away wins can be seen as longshots, but this does not hold for a large portion of the matches.
- **Most-likely/least-likely bias:** Constantinou and Fenton [2013] outline how the odds are tailored in favor of the most-likely outcome of a match. The cumulative loss when betting on all least-likely outcomes was significantly higher than when betting on all most-likely outcomes. Overall, the results appear to be in agreement with the favorite-longshot bias, whereby most-likely bets generate considerably higher returns than least-likely bets.

2.5 Expected goals

There are several companies that gather massive amounts of statistics from football matches. Opta Sports is one of these companies, and by gathering these vast amounts of data, several new statistical measurements which can denote teams strengths and weaknesses have emerged. One of these metrics is called *expected goals* (xG), and it allows you to evaluate team and player performance. Historically, a very popular metric for evaluating which team has produced the most chances, and whether the result of a game is accurate compared to these chances, is the number of shots each team has attempted. One merely counts scoring attempts while the only differentiation among them is whether an attempt went on or off target. Several researchers have used these metrics to measure a team’s attacking strength. Shots and shots on goal may give an indication of how much a team is attacking, but it is not necessarily a metric which represents how many goals each team should have scored, or how many chances they have created. Pollard et al. [2004] reported that scoring attempts differ from each other by a number of parameters. They used data from the World Cups of 1986 and 2002 and investigated the effect of the location of the shot on the chances of scoring, together

with the influence of the following factors:

- The distance from the goal
- The angle from the nearest goalpost
- Distance to the nearest opponent at the time of the shot.

An overview of how the angle to the nearest goalpost and the distance from the goal are calculated can be seen in Figure 2.7, which also depicts the average number of shots per goal from the different shot zones.

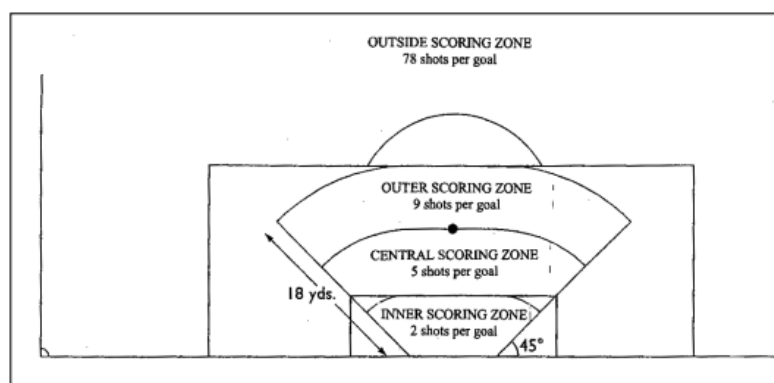


Figure 2.7: Scoring zone with average number of shots per goal from different locations. Taken from Pollard et al. [2004]

Football data experts at Opta have later expanded this model by analyzing over 300,000 shots to calculate the likelihood of an attempt resulting in a goal given a specific position on the pitch. This fairly new metric allows prediction models to more accurately describe each team's attacking strengths, since they now contain more precise and comprehensive data. The idea is that xG's values give an indication of whether a team's results are based on sustainable factors like consistently creating chances or denying the opposition chances, or whether it is down to less sustainable factors like high chance conversion or a sensational goalkeeper. xG is a much better evaluator than purely the shot-on-goal-counter, since shots on goal do not discriminate between a 40-yard shot and an open goal miss.

Related Work

In this chapter we will present related work in regards to football match outcome prediction models and how some of these have inspired us to create our own models, as well as highlighting different money management strategies which can utilize the output from the prediction models in order to make a profit on the betting market. To get a thorough review of the state of the art, one can read our specialization report (Nielsen and Sandøy [2018]).

3.1 Previous match outcome prediction models

This section presents prediction models of football match outcomes which were found in the literature. We will highlight the models in which we have drawn inspiration from when creating our own prediction models.

3.1.1 pi-football

Constantinou et al. [2012] present a Bayesian network model for predicting football games, which they call *pi-football*. Their model considers the factors *strength*, *form*, *psychology* and *fatigue* for both the home and away team. As these factors can be hard to quantify from the available sports data, data-driven objective components are combined with components based on subjective input from a person familiar with the league to construct a complete model.

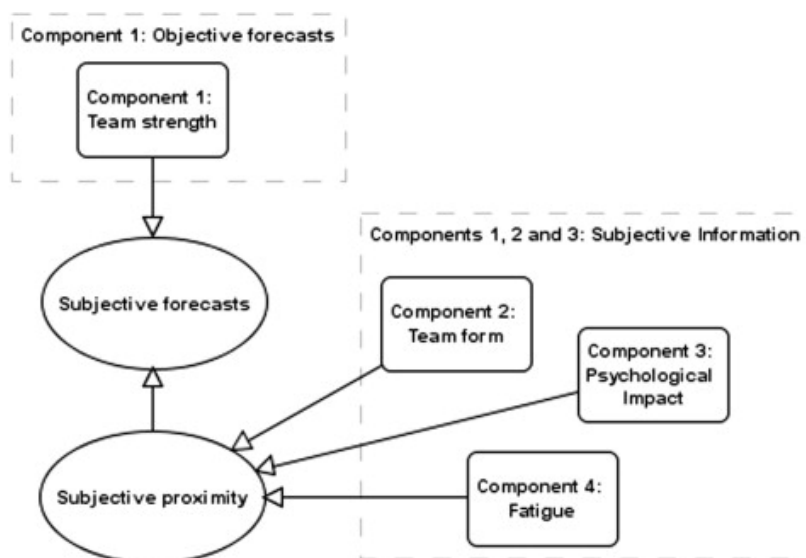


Figure 3.1: Overview of pi-football, from Constantinou et al. [2012]

The objective component models a team’s strength by the points accumulated in the previous five seasons, the team’s current points, and points expected from residual matches in the season. The output of this component is a probability distribution over the three possible outcomes of a football match, which the subjective components can skew towards home or away victory.

Three separate subjective components model each of the remaining factors. Team form is modeled by the availability of three key players specifically, and also the availability of the rest of the first team players as a whole. The impact of first team players returning after being absent a few matches, and whether the team is playing at home or away is also part of the form model. The model for the psychological factor takes into account the impact of the manager, the team’s motivation and team spirit, and potential head-to-head biases. The variables in the fatigue model are the number of days since the last match, first team players rested during last match, the toughness of previous match and the level of national team participation of the team’s players.

In each subjective component, their output is the difference in the given factor between the home and away team. The three outputs together are then used to skew the output of the objective component towards the team it considers has an advantage, resulting in the final subjective forecast. This structure is presented in Figure 3.1.

In evaluating the accuracy of their system, they found that their objective component (i.e. team strength only) was significantly worse than the bookmakers' predictions, but by revising their objective prediction using their subjective components (i.e. form, psychology, fatigue), the system as a whole performed on par with the bookmakers. A profitability measurement was also conducted, wherein their system was evaluated over a complete season of the English Premier League. Three simulations were run, using the highest available odds, the mean odds and the odds of the most common English bookmaker, William Hill. In all three simulations, the systems made a profit, ranging from 2.87% for the mean odds simulation to 9.48% for the William Hill simulation.

A revised version of this system was presented in Constantinou et al. [2013], which both simplified the original system, as well as improving its forecasting capability. In the revised system, the factors *fatigue* and *psychology* are modeled together in a new component called *Fatigue and Motivation*, and *form* is not dependent on identifying three key players as in the original system. Another key change is the way the subjective components are used to influence the match forecasts. In Constantinou et al. [2012], the values of the subjective components for each team were compared, and the discrepancy between them was used to skew the forecast towards either home or away win. In Constantinou et al. [2013], the subjective components are instead used to modify a team's strength distribution, and from the difference between the two teams' distribution, a new forecast can be made. Also, the impact of each model component is inferred hierarchically, and a forecast can be made on each level. The architecture of this revised system is shown in Figure 3.2.

In evaluating this revised system, the authors found that it improved upon the forecasting probabilities of their original system. The forecasts generated at level 2, i.e. considering team strength and a subjective measure of form, were superior to the forecasts at level 1, i.e. considering only team strength from objective historical data. At level 3, the forecasts were inferior to level 2, but still superior to level 1. A sub-component evaluation showed that their model overestimated the negative impact of fatigue, and that this should be taken into account for future models. The authors also suggest that greater results could be achieved by input from a genuine expert, as the results in this paper were achieved by input from a football fan, but not an expert of the league.

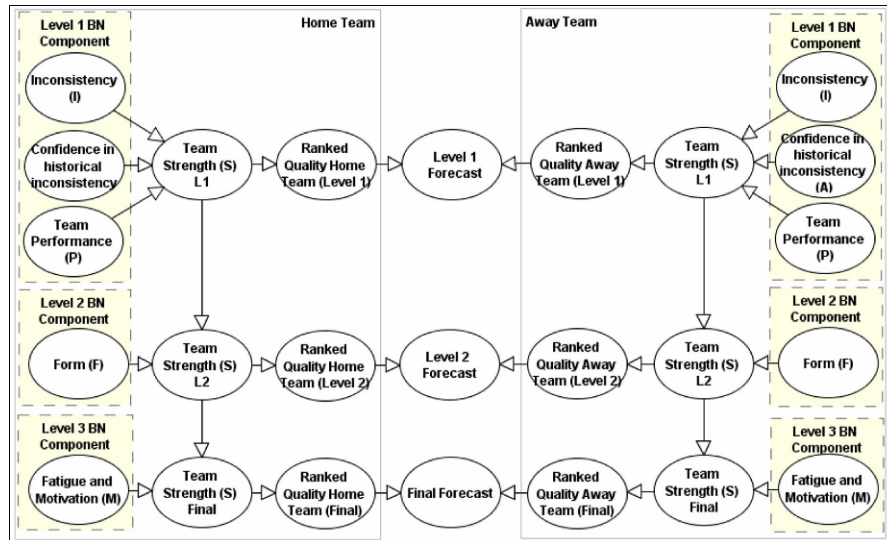


Figure 3.2: Simplified topology of revised pi-football, from Constantinou et al. [2013]

3.1.2 FiveThirtyEight's Club Soccer Prediction

FiveThirtyEight [2018]'s Soccer Power Index (SPI) is an international and club rating system designed to be the best possible representation of a team's current overall skill level. This rating system is a substantially revised version of ESPN's Soccer Power Index, which dates back to 2009 (Silver [2009]). Its goal is to be accurate, reliable and up-to-date. Their system calculates which teams in a given league that are most likely to finish at given table positions. SPI is defined as being a *forward-looking system*. SPI is trying to predict future events, and it uses statistics that correlate better future success, which is the scoring margin. This is an alternative approach compared to other standard models, for example the FIFA rankings, which are based on wins and losses only.

In FiveThirtyEight's Club Soccer Prediction, each team has an offensive rating that represents the number of goals it would be expected to score against an average team on a neutral field, and also a defensive rating represents the number of goals they are expected to concede. These ratings are then used to calculate win/loss/draw probabilities for future matches and simulate the season thousands of times to estimate each team's chances of winning the title. The overall SPI rating is the percentage of points the team is expected to take in a match such as described above.

FiveThirtyEight's Club Soccer Prediction uses three metrics when they are evaluating a team's performance after each match:

- *Adjusted goals*
- *Shot-based expected goals*
- *Non-shot expected goals*

Adjusted goals accounts for the conditions under which each goal was scored. They reduce the value of goals scored when a team has more players on the field, as well as goals scored late in the match when already leading. After down-weighting these goals, they increase the value of all other goals to make the total number of adjusted goals generally add up to the total number of actual goals scored over time.

Shot-based expected goals is equal to the expected goals described in Section 2.5, which is an estimate of how many goals a team should have scored, given the attempted shots.

Non-shot expected goals, however, is an estimate of how many goals a team should have scored based on *non-shooting* actions they took around the opposing team's goals: passes, interceptions, take-ons and tackles.

Both these expected goals metrics have a dynamic score linked to them based on which player is taking the shot, or making the pass. There is an adjustment for each action based on the success rates of the player doing the action, so the fact that Lionel Messi is more probable to score at a shot from 10 yards than Robert Huth is implemented into the model.

When generating the total offensive and defensive score for each team, these three metrics, which are directly comparable, is an average across all three performance metrics.

For each upcoming game in a given league, FiveThirtyEight provides a probability distribution for the match outcomes, which is calculated from the SPI-ratings. See Figure 3.3 for a detailed example.

The model is very dynamic, and as a season plays out a team's rating is adjusted after every match based on its performance in that match and the strength of its opponent. This ensures that a team's rating will not necessarily improve after a win - if it performs worse than expected, its ratings can decline.



1/15	 Everton 4	22%	25%
	 Man. City 0	54%	
		EVE	MNC
	Adjusted goals	3.5	0.0
	Shot-based xG	0.4	0.7
	Non-shot xG	0.7	2.7

Figure 3.3: Performance metrics, prior probabilities and given result between Everton and Man. City in January 2017. Taken from FiveThirtyEight [2018]

As seen in Figure 3.3 it is not always the team that creates the most chances that wins the match. In January 2017 Everton beat Manchester City 4-0, which is a pretty dominant win. Prior to the match, the SPI predicted that Man. City were 54% likely to win the match given the respective team SPI-ratings. Despite the fact that Everton won by a four goal margin, the in-game stats provided by SPI shows that this win may not have been as dominant as one would think. Everton produced fewer chances both based on the shot-based xG and the non-shot xG. This displays how hard it is to create an entirely accurate predictor in the game of football given only statistical approaches. Some minor errors in the predictions must be allowed however, given the sheer complexity of the game. What is important is the fact that the system works in the long haul and can see past some minor fluctuated results.

3.1.3 Beating the Bookmakers

Borøy-Johnsen [2017] used artificial neural networks in order to predict the outcomes of football matches. By collecting a vast amount of relevant data from the football statistics website whoscored.com and mapping these data into different kinds of input patterns, ANNs were constructed in order to predict the outcomes of matches from two successive seasons of the English Premier League. The goal of the thesis was to generate a profit given odds from seven international bookmakers, and the predictions from the neural networks were used to decide whether or not the place bets on the outcomes.

Several different models were proposed and evaluated, using different network topologies, activation functions and input features. Historical football data from whoscored.com is used as input data, and the neural networks try to learn patterns in these datasets. For each prediction model the author tests several network configurations and stores the most promising configuration of each network, which is then used for evaluation in the test cases.

Borøy-Johnsen [2017] showed in his thesis that some of the models created were in fact able to generate a profit across two full seasons of the English Premier League. We highlighted all the models he created in our Specialization Project, and in the upcoming section we will highlight the one relevant to our thesis.

Player Rating

Whoscored.com provides ratings on players which measures how a player contributes to his team. Borøy-Johnsen [2017] proved that these ratings have predictive results by feeding these ratings to a feed-forward-network and predicted the results with 90% accuracy. As input to the prediction model each player's last three matches are taken into consideration, and for each match the following features are added as input:

- *Final rating.*
- *Portion of the match played.* Number of minutes divided by 120. This to increase the impact of players playing the whole match.
- *The player's team's final rating.* Added to capture cases where the player rating are affected by the team's collective effort.
- *Other team's final rating.* Added to capture cases where the player rating are affected by the other team's collective effort.
- *A playing at home-flag.* Added to take home ground advantage into consideration.
- *Days since last match.* Modelled as $e^{\frac{-days-since-match}{7}}$. Added to increase the impact of recent matches.

In addition to these match features, the number of matches for each player the last two game weeks was added as input. This was added to capture possible fatigue

for players. This gives $3 * 6 + 1 = 19$ features for each player. With 22 players participating in each match, this gives $22 * 19 = 418$ features per match.

For the player rating-prediction model a network structure with a single hidden layer of 256 nodes and the *tanh-function* as activation function yielded the most promising results. However, the network did not achieve consistently good results, and combined with several different betting strategies, some of them only generated minor profit and others went completely bankrupt. Borøy-Johnsen [2017] concluded that player ratings tend to overestimate the probability of too many high-odds outcomes. Over the two seasons, the prediction model won approximately 20.4% of all bets places, with an average odds of 4.34.

The most profitable strategy for this model generated a mean ROI of 0.25 in the 2015/2016 season, while the most profitable strategy in the 2016/2017 season generated a ROI of -0.053. The strategy that got a ROI of 0.25 in the first season got a ROI of -1.0 in the second season, which shows the inconsistency of the model.

3.2 Money Management

When applying football prediction models in the field of sports betting the focus shifts from measuring model-accuracy to measuring the profit the model makes. The ultimate goal is not that the model is right every time, but the focus is rather to win money. Whether or not the model makes a profit is determined by a combination of the accuracy of the model and the type of betting strategy the betting agent implements. There are multiple established money management strategies in the field of sports betting, and in the upcoming section we highlight the most popular strategies from the literature.

Money management in the field of sports betting consists of finding *feasible* bets where the odds offered by a bookie is better than the calculated probability for a given outcome implies it should be. In essence this means finding bets in which the prediction models implies a higher probability than the bookmakers' reward. Langseth [2013] outlines that the expected gain per unit at stake is calculated as $P * d - 1$, where P is the calculated probability for that outcome and d is the odds offered. During a round of games, there may be several bets that have a positive expected profit. The punter must then be able to balance how much money to put on each of these bets.

Given a series of expected profitable matches, the issue now consists of deciding on how much money to place on each match. Consider the following examples:

- **Game A:** $P_2 = 0.3, d_2 = 3.6$
- **Game B:** $P_0 = 0.9, d_0 = 1.2$

Here, the expected gain per unit stake for each of the bets is $P * d - 1 = 0.08$. Thus, the games can be seen as equivalent from a punter's perspective, but it is still not clear how much money the punter should put on each of these bets. One can argue that the safest action to take is to stake your money on Game B, since the probability of winning is higher. A downside of this, however, is the fact that one must wager a lot more money on this game than Game A to win the same amount of money, so the risk is higher for Game B. This gives rise to a number of different strategies for so called *money management* problems. Langseth [2013] presents five different money management strategies which determines how much money to place on a bet in a given match, based on the probability P_i of the outcome, the odds d_i , and the bankroll C of the punter. Each of the strategies outputs the amount c_i to place on a given bet. The strategies are as follows:

- **Fixed bet:** A simple strategy of allocating the same amount to each feasible bet, $c_i \propto 1$.
- **Fixed return:** Make sure the same *profit* can be obtained from each bet. This will result in lower amounts staked on the high-gain/low-probability outcomes. $c_i \propto \frac{1}{d_i}$
- **Kelly ratio:** Kelly [1956] suggested a strategy based on a decision-theoretic approach to the money management problem. In this strategy the utility of having an amount C after a bet has been rewarded is defined to be $\ln C$, and thus the utility of going broke approaches $-\infty$. The expected utility of a bet c_i , when the bankroll is C is thus $P_i * \ln(C + d_i c_i) + (1 - P_i) * \ln(C - c_i)$. The utility is maximized by choosing $c_i = C * \frac{P_i d_i - 1}{d_i + 1}$. Langseth [2013] also proposes a modified approach so that the total bets during one round cannot exceed a predefined value C_0 , which is chosen to be much smaller than the bankroll at the beginning of the simulation. This is to ensure that a system that loses heavily during the first few rounds is not excessively punished at a later stage.
- **Variance-adjusted:** Rue and Salvesen [2000] looked at the difference between the expected profit and the variance of that profit, and wanted to minimize this number. After betting c_i , the difference is $P_i d_i c_i - P_i (1 - P_i) (d_i c_i)^2$ which is minimized by choosing $c_i \leftarrow (2d_i(1 - P_i))^{-1}$

- **Markowitz portfolio management:** The variance-adjusted approach can be seen as a simplification of Markowitz portfolio management (Markowitz [1952]). In this strategy one looks at a collection of bets over a game-week. The goal is to find the allocation of bets which maximizes

$$\sum_{i=1}^n (E[\Delta_i] - v \text{Var}[\Delta_i])$$

under the constraint that the bets during the game-week sum to a predefined value C_0 . v represents the accepted level of risk by the punter. The dual representation of the optimization problem is then to minimize $\sum_{i=1}^n \text{Var}[\Delta_i]$ under the constraints $\sum_{i=1}^n c_i = C_0$ and $\sum_{i=1}^n E[\Delta_i] = \mu$. μ a risk acceptance parameter. Langseth [2013] experimented with three different μ -values in his paper:

1. $\mu = \mu_{\downarrow} = (\sum_{i=1}^n P_i d_i) / n - 1$
2. $\mu = \mu^{\uparrow} = \max_i P_i d_i - 1$
3. $\mu = (\mu_{\downarrow} + \mu^{\uparrow}) / 2$

μ^{\uparrow} models the risk-seeking approach, which will force all stakes to be placed on the single bet with the highest expected return. μ_{\downarrow} models a more risk-averse approach, where it only requires the expected return per unit stake of the combined bet to attain the average value of each bet. (Langseth [2013]).

Langseth [2013] compares different statistical models for predicting the outcome of football matches in his paper. He compares three different models by simulating bets being made on matches in the English Premier League using the different betting strategies presented above. Langseth [2013] conducts his experiments over the course of two consecutive seasons of the English Premier League, and in the first season each of the prediction models was able to make a profit using the different money management strategies. It must be noted that the three prediction models experienced its most profitable results using different betting strategies. For the second season, however, only a single combination of prediction model and betting strategy made a profit (this profit was of only 0.4%). These results show how important it is that one chooses a betting strategy which suits the prediction model.

Langseth [2013] argues that one should try to make models that incorporate more of the available information describing the games to improve predictions which in turn could help to beat the bookies.

It should be noted that Langseth [2013] conducted his experiment against the odds from William Hill, which had an average margin of 6.1% during these seasons. This means that losing less than 6.1% should be considered a decent result for a given prediction model.

Chapter 4

Models

In the fall of 2018, we conducted a Specialization Project which investigated the state-of-the-art in predicting the outcomes of football matches based on information available before kick-off (Nielsen and Sandøy [2018]). Based on the collected research, two new prediction models were suggested as possible improvements to the previous models. This Chapter presents these models, as well as the rationale behind them.

In addition to the prediction models, preliminary experiments using reinforcement learning methods to learn a money management system were performed. This Chapter also presents these experiments.

4.1 Data-driven pi-football

The two different versions of a Bayesian network-based model called *pi-football*, presented in Constantinou et al. [2012] and Constantinou et al. [2013], were both able to achieve a positive return on investment over a full season of the English Premier League. The downside of their models, however, is that they rely on subjective input from an expert of the league. The core idea of pi-football is to attempt to model the four generic factors *strength*, *form*, *psychology* and *fatigue*, and their impact on the outcome of a football match. Of these factors, only the strength factor is modeled by objective historical data, while the rest are dependent on the expert's input.

While pi-football's promising results could be attributed to the use of expert knowledge, the four generic factors could possibly also be modeled by historical data. The advantage of going for a purely data-driven approach is that the same model could be applied to different leagues, as long as the data is available. It would also remove any subjective biases an expert may have.

Our model tries to utilize some key features taken from both pi-football and FiveThirtyEight, described in Section Section 3.1.1 and 3.1.2 respectively. The data set from FiveThirtyEight offers a wide range of interesting data. Data points such as expected goals, SoccerPowerIndex-rating and match importance for each team are all data which could help eliminate the need for a subjective input in the pi-football system. FiveThirtyEight claim that their system has powerful predictive power, and we want to test this predictive power in our thesis by feeding the available data from FiveThirtyEight as input features into an artificial neural network.

A challenge of a data-driven approach without the help from a domain expert, could be to identify data points that may influence the slightly abstract factors *form, psychology and fatigue*. Given FiveThirtyEight's vast and granular dataset, described in Section 5.1.2, an opportunity to eliminate the subjective input from the pi-football-model has arrived.

An overview of the input features to the model is presented below.

Input

- **Strength factors**

- *SoccerPowerIndex-rating for home team.*
- *SoccerPowerIndex-rating for away team.*

- **Form factors**

- *Form-aggregated SoccerPowerIndex-rating for home team.*
- *Form-aggregated SoccerPowerIndex-rating for away team.*

- **Psychology factors**

- *Match importance for home team.*
- *Match importance for away team.*

- **Fatigue factors**

- *Days since last match for home team.* Modeled as $e^{\frac{-days-since-match}{7}}$. Added to increase the impact of recent matches.
- *Days since last match for away team.* Modeled as $e^{\frac{-days-since-match}{7}}$. Added to increase the impact of recent matches.

This yields 8 different features fed into the network as input per match. For the strength factor, FiveThirtyEight’s SoccerPowerIndex may work perfectly to measure a team’s strength. As described in Section 3.1.2, the overall SPI rating is the percentage of points the team is expected to take in a match against an average team on a neutral field.

To model a team’s form, the data-driven pi-football system applies the *exponentially moving average-formula* to the SPI-rating for each team. Exponentially moving average is a first-order infinite impulse response filter that applies weighting factors which decrease exponentially (Investopedia [2019]). In short, an EMA is like a simple moving average, except it weights recent instances more than older instances based on an alpha parameter. This allows the calculated rating to put more weight on recent ratings and less on past rating, which in turn models the team’s current form.

The match importance calculated for both teams by FiveThirtyEight could be a perfect way to model the psychology factor for the pi-football-system. Match importance is a measure of how much the the outcome of the match will change each team’s statistical outlook on the season, and this can be seen as a potential replacement of the need for a domain expert offering input to the Bayesian model.

Fatigue is modeled by counting the number of days since the last match for each team and applying the exponential function $e^{\frac{-days-since-match}{7}}$ to increase the impact of more recent matches.

4.2 Expected goals of starting lineup

One of the most promising papers discussed in our Specialization Project (Nielsen and Sandøy [2018]) is Borøy-Johnsen [2017], who used artificial neural networks as an architecture for his prediction models. Borøy-Johnsen [2017] constructed several different prediction models using neural network and was able to generate a profit over two full seasons for some of them. One model which had an interesting rationale was the Player Rating-network, described in Section 3.1.3. The idea was

to capture how much each player in a starting lineup contributes to his team in that given match. Borøy-Johnsen [2017] attempted this by applying the player ratings scraped from whoscored.com. He also used each team's total rating as input features to the network. This model failed to capture consistent results, yielding a profit for one of the test seasons, but a loss for the other. These results indicate that these ratings did not have the predictive power to accurately estimate the outcomes in football matches.

A better approach would be to apply player statistics which have a greater impact on the outcome of the match. In this regard, utilizing player level expected goals from Understat, described in Section ??, as input features instead could prove to be a better predictor for football outcomes. This metric describes how much each player in a team has contributed to the number of goals his team is expected to score, which is a statistic that directly impacts what we are trying to predict. By scraping understat.com and storing the expected goals-values for each player in every available match in a database, the total expected goal for a team's lineup can be calculated prior to that game.

By taking another approach than Borøy-Johnsen [2017] for the Player Rating model, but still using the days since last match-feature, we want to test this modification to the model, and test if this improves the model and can produce more consistent results. Borøy-Johnsen [2017]'s Player Rating-model was implemented by only taking the three latest matches a team has played into account. This essentially means that if a reasonably strong team has played stronger opposition in the latest matches, it will be regarded as a relatively weak team in the next match. We will try to battle this problem by implementing an expected goals-value based on current form and also an expected goals-value based on the team's performance for the whole season. By incorporating team performances for the whole season and also having a recent matches-factor, both the strength factor and the form factor for teams are persisted as features to the network.

Since expected goals mostly depict the attacking strength of a team, our model would fail to capture the defensive strengths of teams if only xG-values would be fed into the model. To battle this, the model will consider the expected goals against(xGA)-values provided by Understat. This data represents the defensive strength of a team since it is a measure of how many goals a team is expected to concede. By considering both the current form of the team for the last five matches, and also the overall quality of the team during the whole season in the same approach as the expected goals-values, our model will try to incorporate the defensive strengths of each competing team.

An overview of the input features to the model is presented below.

Input

- *Form-aggregated xG90 for each player in the home team.* Taking the last 5 games into consideration.
- *Form-aggregated xG90 for each player in the away team.* Taking the last 5 games into consideration.
- *Current season-aggregated xG90 for each player in the home team.* Accumulating the offensive form of the player over the whole season.
- *Current season-aggregated xG90 for each player in the away team.* Accumulating the offensive form of the player over the whole season.
- *Form-aggregated xGA90 for the home team.* Taking the last 5 games into consideration.
- *Form-aggregated xGA90 for the away team.* Taking the last 5 games into consideration.
- *Current season-aggregated xGA90 for the home team.* Accumulating the defensive form of the team over the whole season.
- *Current season-aggregated xGA90 for the away team.* Accumulating the defensive form of the team over the whole season.
- *Days since last match for the home team.* Modeled as $e^{\frac{-days-since-match}{7}}$. Added to increase the impact of recent matches.
- *Days since last match for the away team.* Modeled as $e^{\frac{-days-since-match}{7}}$. Added to increase the impact of recent matches.

There are 22 players in the starting lineups, each with 2 features each. This gives $22 * 2 + 2 + 2 + 2 = 50$ different features fed into the network per match. Both offensive and defensive team-strength is modeled in the network, with the expected goals and expected goals against-values respectively. Form is modeled by attempting to capture the current form of each player by taking the last 5 games into consideration and aggregating expected goals and expected goals against-values. Fatigue is modeled by considering how many days its been since each team played its last game.

4.3 Network Structure

As described in Section 1.1, our thesis is investigating whether or not utilizing domain knowledge when structuring artificial neural networks increases their predictive performance. This particular research question is inspired by the pi-football-model (Constantinou et al. [2012], which groups related features into different components in a Bayesian Network. By grouping the features related to each component together in different sub-networks in a neural networks architecture before merging the sub-networks into a single network will isolate them from the rest of the features for a number of layers, which may allow the network to find valuable intermediary representations, before merging the layers later on treats the problem as a whole.

Thus, for both of our suggested prediction models, we will attempt to group related features together and isolating these features in individual sub-networks before grouping them together. See Figure 4.1 for an overview of such a composition. In this example the features are grouped into four different components: strength, form, psychology and fatigue. Figure 4.1 also shows the identical features applied to a fully connected neural network instead. In this setup, every input feature is connected to every weight in the corresponding hidden layer. By doing this kind of grouping and measuring the performances of the different networks our thesis will have a solid basis to answer the constructed research question.

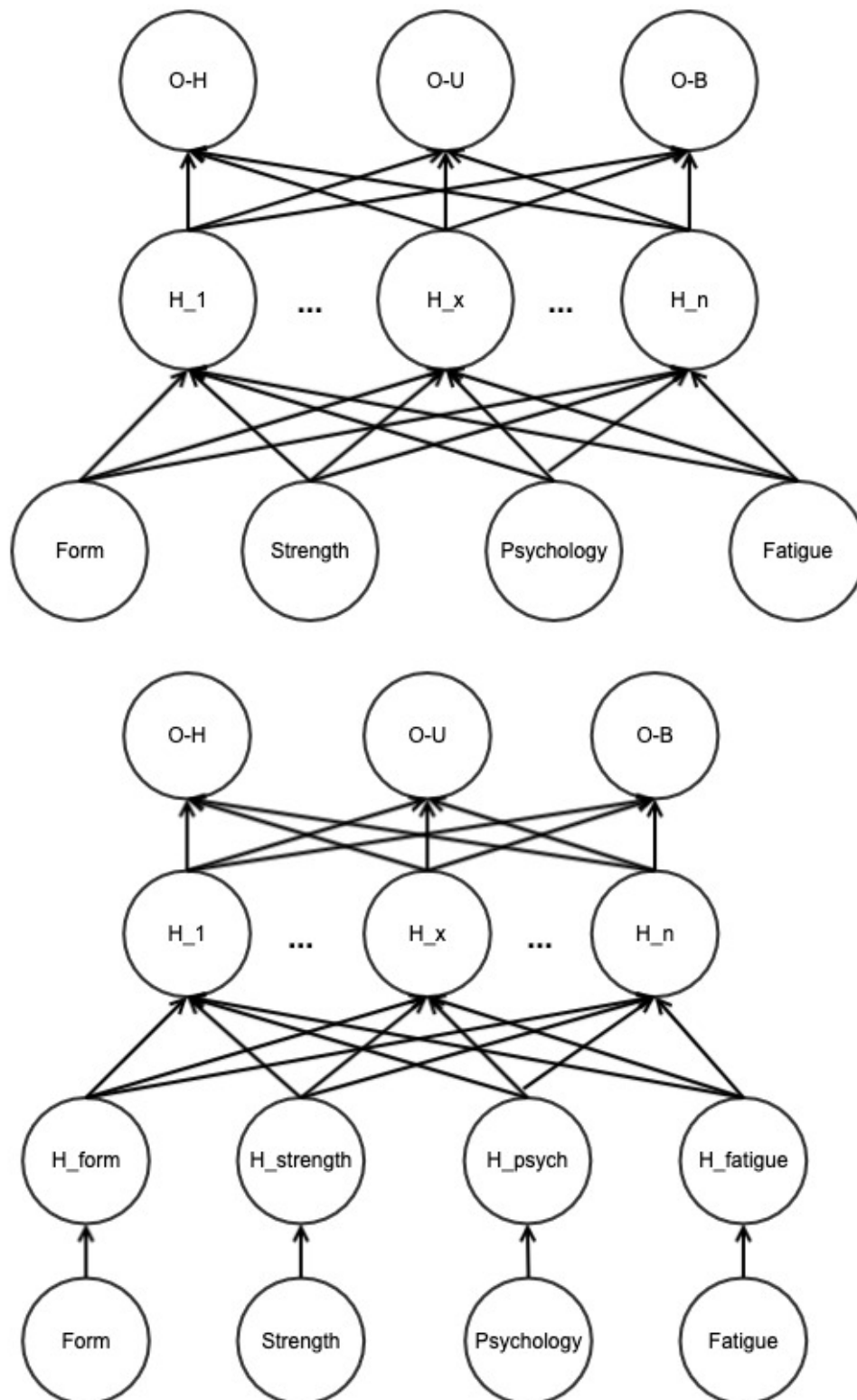


Figure 4.1: The two different network architectures. The top figure displays a fully connected model with four different components. The bottom displays the sub-network-architecture, which groups the components into sub-networks before merging them.

During this thesis we aim to investigate whether or not this kind of grouping using domain knowledge increases the predictive performance of the network, and our two suggested prediction models will be constructed in a fully connected- and a sub-network fashion. In total, this means that 4 different models will be trained and tested in our experiment.

4.3.1 Model component grouping

The component grouping for the pi-football model is described in Section 4.1, and the model is split up into four different components: strength, form, psychology and fatigue.

Since each model has a unique set of features, this makes it difficult to group the models into a fixed set of components based on these features. The psychology component which the pi-football model utilizes is not incorporated into the expected goals of starting lineup model, and thus this component is excluded in the model.

An overview of the grouping of the input features for the expected goals of starting lineup model is presented below.

Expected goals of starting lineup

- **Strength factors**

- *Current season-aggregated $xG90$ for each player in the home team.*
- *Current season-aggregated $xG90$ for each player in the away team.*
- *Current season-aggregated $xGA90$ for home team.*
- *Current season-aggregated $xGA90$ for away team.*

- **Form factors**

- *Form-aggregated $xG90$ for each player in the home team.*
- *Form-aggregated $xG90$ for each player in the away team.*
- *Form-aggregated $xGA90$ for home team.*
- *Form-aggregated $xGA90$ for away team.*

- **Fatigue factors**

- *Days since last match for home team.*
- *Days since last match for away team.*

The rationale behind this grouping is relatively straightforward. The fatigue factor is grouped in the same way as pi-football as this is the identical data-point. For the strength factor the defensive and offensive performances for the team is grouped together and bound by the performances across the whole season, while the form factors takes the same data-points but the data is for a smaller window.

4.4 Money Management

A number of experiments were performed to explore the viability of using reinforcement learning techniques in a money management context. Given a prediction model’s forecasts and a bookmaker’s odds, an agent is to learn to place profitable bets on an upcoming football match. For a match with true probabilities $\{P_h, P_d, P_a\}$ and odds $\{d_h, d_d, d_a\}$ (h , d and a meaning home team win, draw and away team win), placing a unit bet on an outcome $o \in \{h, d, a\}$ where $P_o * d_o > 1$ would be a rational action, as the expected value of the bet would be greater than zero. The problem is of course that we do not know the true probabilities, and treating the softmax output of a prediction model as probabilities may also be problematic, as the softmax output cannot be accurately interpreted as the model’s confidence (Gal [2016]). If an agent could learn a relationship between the outputs and the model’s real confidence, it may be able to find a different threshold for making rational actions.

A reinforcement learning agent learns to adapt to its environment, and care must be taken when modeling the environment to achieve the desired behavior in the agent. In our model of the environment, a time step represents a single football match, and the fundamental elements of the observation space are

- *Outcome predictions*, i.e. predicted probabilities of home team victory, away team victory or draw,
- *Match odds* from a single bookmaker, and
- The agent’s *current bankroll*.

When betting on the outcome of a football match, there are four distinct classes of actions to take: bet on home team, bet on away team, bet on draw, or refrain

from betting. One must also decide how much one should wager, in the range $[0, b]$, where b is the current bankroll of the punter, making the action space a subspace of \mathbb{R}^3 , with the origin representing the no-bet action. A simplification of the action space has been made in these experiments, where the wagers are fixed at one unit. This reduces the action space down to the four discrete choices $\{H, D, A, NB\}$, meaning bet on home team, bet on draw, bet on away team, or no bet respectively.

The agents were trained using a Deep Q-Network (Mnih et al. [2015]), a model-free reinforcement learning method which should be a good fit due to the size of the multidimensionally continuous observation space.

4.4.1 Weekly Betting Environment

When betting on a football match, one obviously has to wait for the match to end before knowing how the bet went. In most leagues multiple matches are played simultaneously, and a punter has to decide how to spread his bets before they start, and does not receive his winnings before after the matches are over.

If one is to consider one match at a time, as an agent would do with our model of the environment, a punter would need to keep track of how many more matches he has to consider before payout, as well as how well he expects his already placed bets to pay. In order to capture this in a betting agent, these variables are added to the observation space.

As an agent would not know how his bets would resolve before all the matches have finished, the agent is only rewarded when the last bet in a round has been placed. The reward function $R(s_t, a_t)$ was therefore implemented as

$$R(s_t, a_t) = \begin{cases} -10 & \text{if } b < 0 \\ 0 & \text{if not done with betting round} \\ \sum_{i=0}^n Profit(s_{t-i}, a_{t-i}) & \text{if done with betting round} \end{cases}$$

where n is the number of matches in a betting round, and

$$Profit(s_t, a_t) = \begin{cases} 0 & \text{if } a_t = NB \\ -1 & \text{if lost bet} \\ d_a & \text{if won bet} \end{cases}$$

These experiments were done in parallel with the development of our prediction models, therefore the match forecasts used are from FiveThirtyEight (FiveThirtyEight

[2018]) rather than from our own models. The odds used are the closing odds from William Hill. An agent was trained on 3903 matches from the 2016/2017, 2017/2018 and half of the 2018/2019 seasons of the English Premier League, Bundesliga, Serie A, La Liga and Ligue 1. The matches are randomly drawn from this pool, and an episode ends after the agent has considered 500 matches, or is bankrupt. A betting round was defined to consist of 10 matches, meaning 10 consecutive time steps.

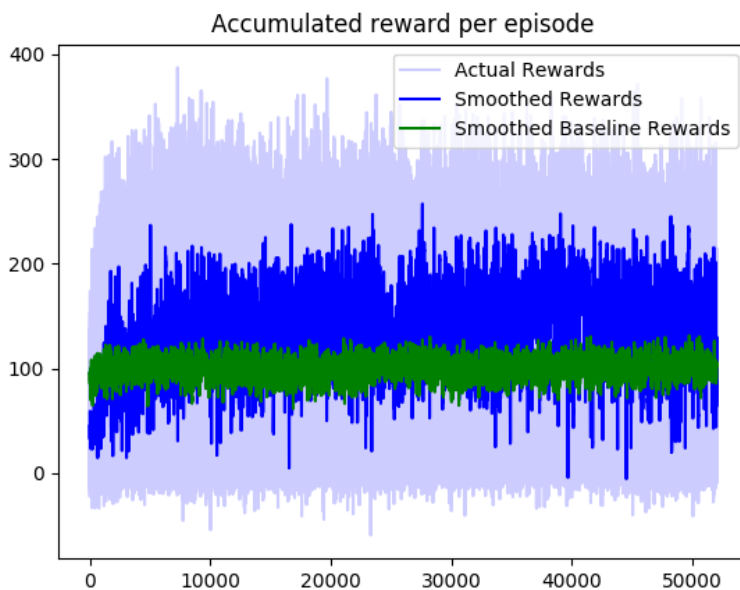


Figure 4.2: Accumulated reward per episode, Weekly Environment

Figure 4.2 shows the development of the reward accumulated by the agent over the training episodes, along with the reward achieved by following a baseline policy $\pi(s) = \underset{a}{\operatorname{argmax}}(P_a * d_a - 1)$ (with $P_{NB} = d_{NB} = 1$). As seen in the figure, the agent fails to improve in any significant way, and the per-episode reward varies wildly.

The policy learned by the agent is to almost always bet on the home team, choosing the H action 96% of the time, with the remainder being the no-bet action NB and the draw and away team bets never being selected at all. On average, the net profit achieved in test episodes was -3.4%, but the worst case was bankruptcy, and the best case was 272.4%, meaning the policy is very unpredictable in terms of profit. Regardless of profitability, the agent seemingly fails to capture the relationship

between the forecasts and the odds, and the resulting behavior is trivial as a betting strategy.

4.4.2 Perfect Information Environment

As the more realistic approach of weekly betting failed to give satisfactory results, exploratory experiments were performed using a highly stylized environment. In this environment, the match forecasts observed by the agent are the true probability distributions over the outcomes. Additionally, the odds observed are randomly generated and not correlated with the outcome probabilities. This should be a significantly easier environment for the agent to learn, as the bets where $P_o * d_o > 1$ will always be rational. These bets will also be more frequently observed when the odds and probabilities are not correlated. Another simplification of this environment is disregarding the fact that multiple matches are played simultaneously; the agent sees the reward and profit of its action at each time step.

Feasibility Reward

To verify that an agent could make rational decisions with the given observation space, an agent was trained with the following reward function:

$$R(s, a) = \begin{cases} 0 & \text{if } a = NB \\ P_a * d_a - 1 & \text{otherwise} \end{cases}$$

This reward function should teach the agent to bet on outcomes where the expected value of the bet is greater than zero, which in the long run should be a viable betting strategy. While the resulting money management strategy would not be very interesting and easily achieved without using reinforcement learning, this reward function serves to confirm that the environment contains enough information for an agent to learn. As the agent is not exposed to any uncertainty in the outcomes, we expect the agent to learn optimal behaviour quickly, placing almost all bets in the profitable zone.

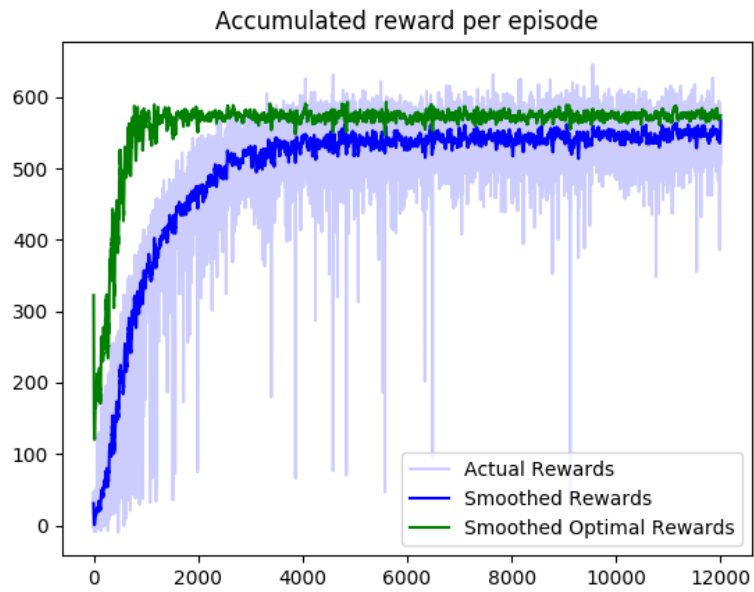


Figure 4.3: Accumulated reward per episode, Perfect Information Environment, Feasibility Reward

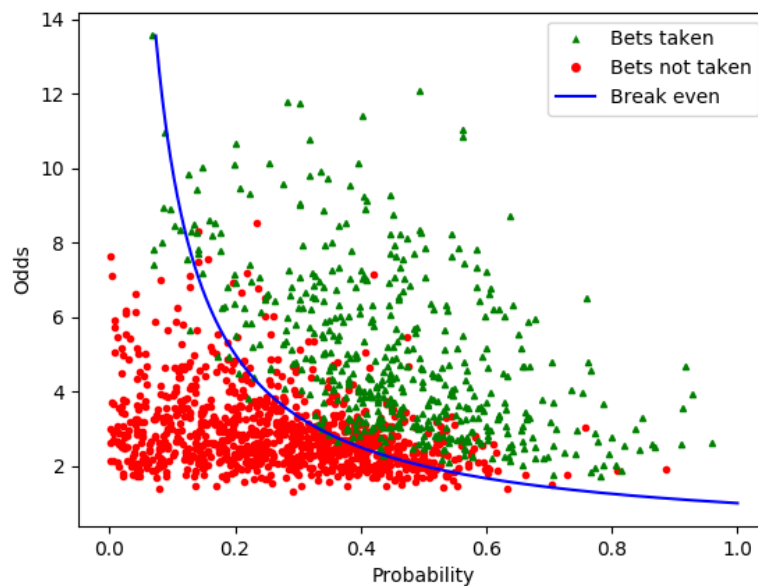


Figure 4.4: Distribution of bets made by agent trained in Perfect Information Environment using Feasibility Reward

As Figure 4.3 shows, the agent is clearly able to learn its reward function. The policy used as a baseline policy in the weekly betting experiment is in this environment the optimal policy, as the probabilities used here is the true probability distribution over the outcomes. As seen in the figure, the agent learns a policy very close to optimal. Figure 4.4 shows the bets considered by the agent during a test episode after training. As we can see from the figure, the agent places almost all its bets in the profitable side of the break-even line, i.e. the area where $P_o * d_o > 1$. We also see that there are quite a few bets in the profitable area that the agent has decided not to take. This is likely due to the agent only placing one bet per match, and since the odds are generated at random, there could be more than one feasible bet available for each match.

Profit Reward

Having verified that recognizing profitable bets is possible, a single-match version of the reward function used in the weekly betting experiment was implemented:

$$R(s, a) = \begin{cases} 0 & \text{if } a = NB \\ -10 & \text{if lost bet and bankroll} < 0 \\ -1 & \text{if lost bet} \\ d_a & \text{if won bet} \end{cases}$$

Since the agent's possible actions is to place a unit bet on an outcome (or don't bet at all), this reward function corresponds to the profit made on the bet. Using this reward function, the agent is actually exposed to the uncertainty in the outcomes. As the agent knows the exact probability distribution over the outcomes, we expect the agent to again learn a similar policy to the previous experiment, i.e. showing a preference for bets where $P_o * d_o > 1$.

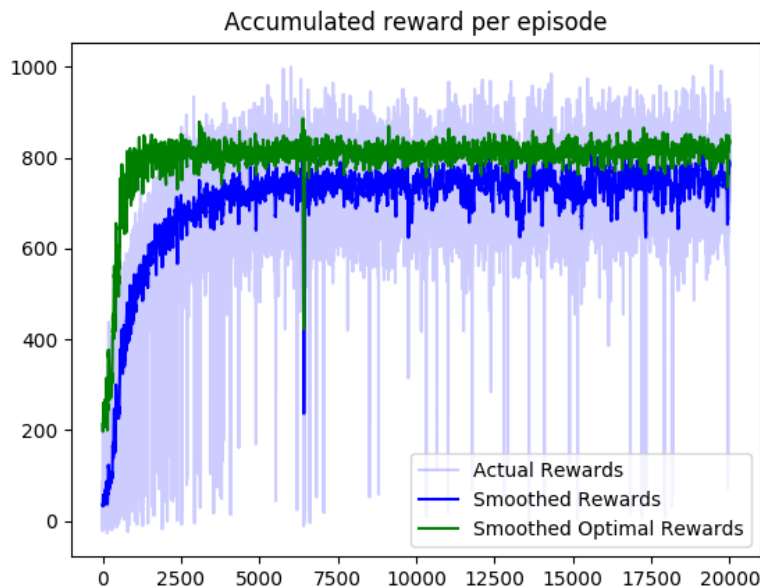


Figure 4.5: Accumulated reward per episode, Perfect Information Environment, Profit Reward

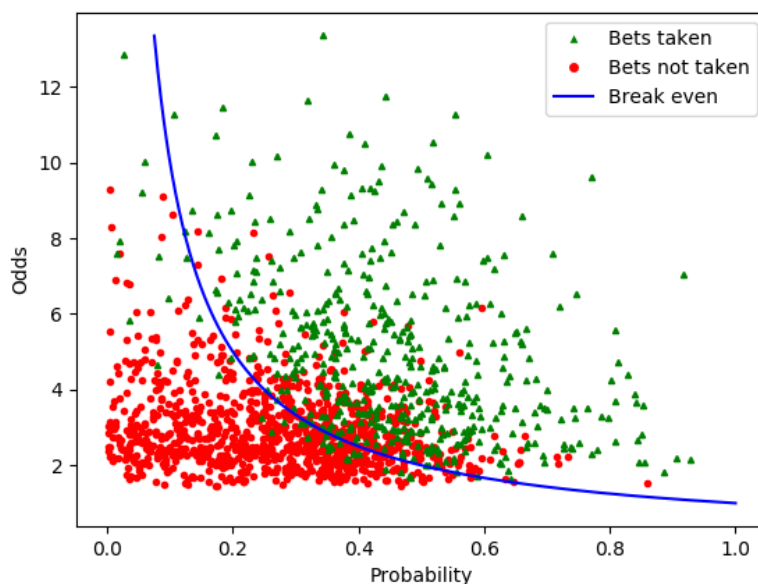


Figure 4.6: Distribution of bets made by agent trained in Perfect Information Environment using Profit Reward

As with the feasibility reward, Figure 4.5 shows that the agent is able to learn using this reward function, but with greater variance in reward from episode to episode. The same optimal policy as in the feasibility experiment is also displayed, and we see that the policy learned by the agent is very close to optimal.

The bet distribution shown in Figure 4.6 is very similar to the one in the previous experiment. This indicates that the agent managed to learn the relationship between the odds and outcome probabilities and makes mostly feasible bets even when faced with random sampling of outcomes.

4.4.3 Single Match Environment

As the experiments using perfect information showed that learning to recognize profitable bets is possible, the logical next step is to reintroduce the real data. This environment uses the same data as in the weekly betting experiment, but using the simplification of treating each match on its own, as in the perfect information environment.

This environment should be a great deal more challenging than the previous one. Not only are we relying on the match forecasts being reasonably accurate, but also for there to be a discrepancy between the forecasts and the implied probability of the odds. If the bookmaker and the predictor are in agreement on the probability distribution of the outcomes, we have $P_o * d_o = 1$, and there is no margin to make a profit. In the previous environment with perfect information, the odds and probabilities were not correlated, giving the agent many opportunities to find bets where the discrepancy between odds and predictions were favorable. This is not likely to be the case with real predictions and odds.

The goal of these experiments is to see if an agent can learn which bets are rational to make given the observed predictions and odds. As we can't assume the predictions are the true probabilities, the rational bets may not be all bets where $P_o * d_o > 1$, but might be along another implicit curve depending on the biases and uncertainty of the predictions. As such, the feasibility reward function is largely uninteresting in this case, and is therefore not used in these experiments.

Profit Reward

An agent was trained in this environment using the same profit reward function used in the Perfect Information environment. Figure 4.7 shows the development of the accumulated reward per episode, along with the reward accumulated by following the policy $\pi(s) = \underset{a}{\operatorname{argmax}}(P_a * d_a - 1)$. This policy - which we described as the optimal policy in the perfect information environment - is here treated as a baseline again, as we cannot assume that the policy is optimal in this environment.

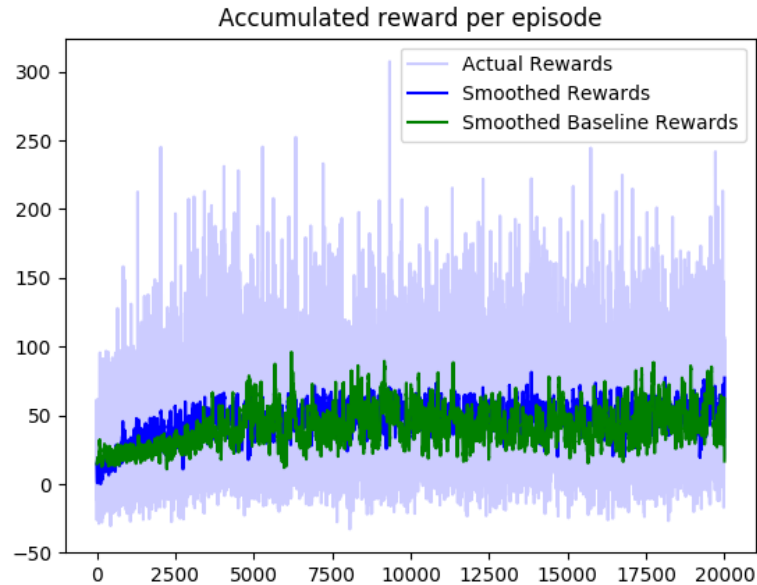


Figure 4.7: Accumulated reward per episode, Real Data Environment, Profit Reward

As seen in the figure, the agent does not improve in any significant way, and there is a high degree of variance in the rewards achieved. It does however fare quite similarly to the baseline policy, which suffers from the same problems.

Figure 4.8 shows the aggregated distribution of bets made by the agent over 100 separate test episodes, where we can see that behavior learned by the agent is to bet on outcomes where P_o is approximately greater than 25-30%, regardless of odds. This is distinctly different from a policy of betting when $P_o * d_o > \tau$ for some $\tau \in \mathbb{R}$ like we had expected.

The agent has mostly learned to avoid betting, choosing the no-bet action around 62% of the time. When it actually does place a bet, the vast majority of the bets made are on the home team, with 30% of the actions chosen. Even though it has adopted a rather conservative behavior, it is not very effective, as on average, the agent is bankrupt after 325 matches, with approximately 70% of episodes ending in bankruptcy. Overall, it seems that the agent does not manage to find an exploitable relationship between the forecasts and odds, and therefore does not learn a useful strategy for profiting on the bets.

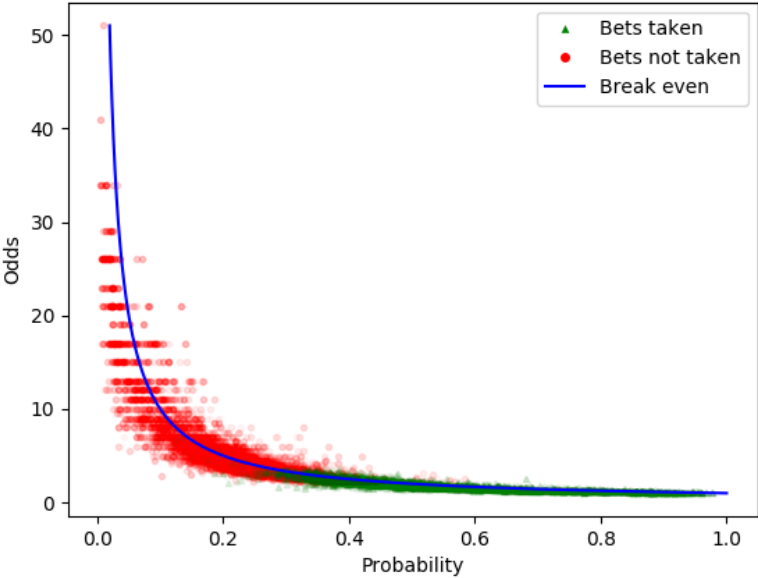


Figure 4.8: Aggregated distribution of bets made by agent trained in Real Data Environment

Data and System Design

A thorough review of several sources of football data was conducted as part of the specialization project preceding this Master project. In this chapter, we will present a condensed version of the Data Sources-chapter from our specialization project (Nielsen and Sandøy [2018]), as well as the general architecture of the components of our system.

5.1 Data sources

5.1.1 Understat

Understat (www.understat.com) is a football statistics website based around the *expected goals* (xG) metric described in Section 2.5, offering detailed statistics and metrics for some of the top European leagues going back to the 2014/2015 season.

For each match, Understat offers data on a *shot* level, a *player* level, and an overall match level. The data consists of both basic football statistics such as goals, shots on goals and bookings, but also their own data on a higher abstraction level, such as expected goals and expected assists. Table 5.1 shows the data points available on the match level, while Table 5.2 shows the data points available for each player in a given match.

Data Point	Description
Date	The date the match was played
Teams	Names of the teams playing
Goals	Goals scored by each team
xG	Expected goals at match end for each team
Shots	Number of shots made
Shots on target	Number of shots made that went on target
Deep passes	Number of passes completed within 20 yards of goal
PPDA	Passes allowed per defensive action in the opposition half
xPTS	Expected points

Table 5.1: Description of match level data from Understat.com

Data Point	Description
Name	Name of the player
Position	Position of the player in the match, e.g. goal keeper, forward
Goals	Number of goals scored
Assists	Number of passes made by the player that lead to a goal
Key Passes	Number of passes made by the player that lead to a shot
Shots	Shots made by the player
Cards	Number of yellow and red cards received
Time	Number of minutes played in the match
Substitution time	Match time when the player was brought on or off
xG	Expected goals of shots made by the player
xA	Sum of xG values of shots from the players key passes
xGChain	Total xG of every possession the player was involved in
xGBuildup	Total xG of every possession the player is involved in without key passes or shots

Table 5.2: Description of player level data from Understat.com

One can for example look up the number of goals Eden Hazard is expected to contribute to during a match, and make assumptions on Chelsea's next performance based on if he is starting the game or not. If Chelsea's xG drops massively because Diego Costa suddenly fell ill before kick off, this might have a great impact on the outcome of the match. Which players starts for a given team in the Premier League is announced an hour before kick off, so if one is using Understat as a data-source combined with the official line-up, one can easily calculate how a team is expected to perform given a certain line-up.

Nº	Player	Pos	Apps	Min	G	A	Sh90	KP90	xG	xA	xG90	xA90
1	Diego Costa	F	35	3101	20	7	3.22	1.22	15.43 ^{-4.57}	4.91 ^{-2.09}	0.45	0.14
2	Eden Hazard	F M	36	3050	16	5	2.27	2.60	11.03 ^{-4.97}	6.44 ^{+1.44}	0.33	0.19
3	Pedro	F M	35	2178	9	9	2.44	1.74	5.99 ^{-3.01}	5.53 ^{-3.47}	0.25	0.23
4	Willian	F M	34	1509	8	2	2.62	3.04	3.76 ^{-4.24}	3.06 ^{+1.06}	0.22	0.18
5	Gary Cahill	D	37	3299	6	0	0.65	0.14	3.30 ^{-2.70}	0.17 ^{+0.17}	0.09	0.00
6	Marcos Alonso	M	31	2696	6	3	1.54	0.80	4.15 ^{-1.85}	3.15 ^{+0.15}	0.14	0.11
7	Cesc Fàbregas	M	29	1294	5	12	1.81	4.31	2.81 ^{-2.18}	7.85 ^{-4.15}	0.20	0.55
8	Michy Batshuayi	F	20	221	5	1	6.92	0.41	3.76 ^{-1.24}	0.03 ^{-0.97}	1.53	0.01
9	Victor Moses	M	34	2519	3	2	1.61	1.00	4.05 ^{+1.05}	2.63 ^{+0.63}	0.14	0.09
10	César Azpillicueta	DM	38	3420	1	4	0.47	0.58	0.64 ^{-0.36}	1.68 ^{-2.32}	0.02	0.04

Figure 5.1: Overview of Chelsea’s player-statistics during the 2016/2017-season and the expected values for each player, taken from Understat.com. The green and red numbers denotes the difference between expected goals and actual goals.

Understat does not offer their data through an API or downloadable files, and can only be viewed by browsing through their website. As such, to efficiently collect all their data, an HTML scraper must be used.

As mentioned, Understat offers detailed statistics and metrics for some of the top European leagues dating back to the 2014/2015-season. These leagues are

- English Premier League
- Italian Serie A
- Spanish La Liga
- French Ligue 1
- German Bundesliga
- Russian Premier League

This is in total 6 different leagues, four of them having 380 matches per season, while the German Bundesliga has 340 matches per season and the Russian Premier League 300 matches per season. Dating back 4 full seasons, this means that Understat.com offers data for $4 * 6 * 380 + 340 + 300 = 9760$ different matches.

5.1.2 FiveThirtyEight

FiveThirtyEight is a website focusing on statistical analysis of politics, economics and sports. In January 2017, they started their Club Soccer Prediction project, where they attempt to predict the outcome of football matches in the top leagues around the world. The forecasts are based on a substantially revised version of ESPN’s Soccer Power Index (SPI), which is a measure of the strength of a team within their league. After each match, the teams’ SPI is adjusted based on three metrics they call *adjusted goals*, *shot-based expected goals* and *non-shot expected goals*. They also calculate the importance of a given match for the two teams, based on how the teams’ position in the league table may shift based on the outcome of the match.

All their forecasts, as well as the underlying metrics and raw data from 2016 and onward is available for download in CSV format from their website.

FiveThirtyEight’s data dates two complete seasons back, which means that data available starting from the 2016/2017-season in the English Premier League is available. See Table 5.1.2 for a detailed description of what kind of data FiveThirtyEight offers.

Data	
Date	Actual #goals Team 1
Team 1	Actual #goals Team 2
Team 2	Match importance Team 1
SPI for Team 1	Match importance Team 2
SPI for Team 2	xG for Team 1
Prob. for Team 1	xG for Team 2
Prob. for Team 2	Non-shot xG for Team 1
Prob. for tie	Non-shot xG for Team 2
Projected #goals Team 1	Adjusted goals for Team 1
Projected #goals Team 2	Adjusted goals for Team 2

Table 5.3: Overview of the data available from Soccer Power Index.

What makes FiveThirtyEight an interesting data source is that they offer data from over 30 different leagues all over the world. Each league has data for two complete seasons back and this means that there is over 60 different seasons for our models to train on. A result of this is that our models will be able to train on more generalized data and not only the data which is available for the English Premier League, which in turn could allow us to extend our models to multiple

different leagues while still achieving a profit.

5.1.3 Football-Data

Football-Data.co.uk is a free football betting portal providing historical results and odds, downloadable in convenient Excel or CSV formats (Football-Data [2018b]). They offer datasets for up to 22 European league divisions from the 1993/94 season and onwards. The datasets up until the 2000/01 season only contain fulltime and halftime results, but from then on, the datasets also contain match betting odds from up to 10 major bookmakers. The 2000/01 season datasets also introduced match statistics such as shots on goal, corners, fouls and more, similar to those available through Understat Football-Data [2018a].

5.2 Data collection

5.2.1 Understat scraper

As Understat does not offer an API or downloadable files, an HTML scraping system was created using Python and the Python library *BeautifulSoup4*. A given season of a league Understat has data on is accessed at `www.understat.com/league/<league_name>/<season>`. At this page, a Javascript variable called `datesData` contains the match IDs for all the matches of the season. This variable is parsed by our scraping system, and used for accessing each match page. On each match page (at `www.understat.com/match/<match_id>`), the three Javascript variables `shotsData`, `rosterData` and `matchData` are parsed by the scraper. These three variables contain the shot level data, the player level data and the overall match data, respectively.

As Understat is the most comprehensive and granular of the data sources used in this project, the data collected from Understat serves as the basis of our database, in terms of team names and match IDs.

5.2.2 Football-Data scraper

Because Football-Data.co.uk offers its data as downloadable files, the data collection procedure for this data source is simpler than for Understat. The data is organized as a CSV file for every season of every league, available at www.football-data.co.uk/mmz4281/<season>/<league_code>.csv. Using the Python library *Pandas*, the CSV files are downloaded, parsed and manipulated as the data is extracted and stored in the database.

5.2.3 FiveThirtyEight

FiveThirtyEight offers its data as downloadable files in a CSV-format. The data is available at

https://projects.fivethirtyeight.com/soccer-api/club/spi_matches.csv.

Using the Python library *Pandas*, the CSV file is downloaded, parsed and manipulated as the data is extracted and stored in the database.

5.3 System Design

5.3.1 Database

In order to store the data available from our data sources, a relational database has been applied. This is in order to gather the data in one place with a relational structure that easily allows our models to collect their necessary data. By storing the data in a database, this saves a lot of time and computer resources since we only have to scrape the data sources once.

Table overview

Figure 5.2 shows an overview of the tables stored in the database. Please note that the attributes stored in the figure only is a bare-minimum representation of the tables and a more thorough overview can be inspected in Appendix A.

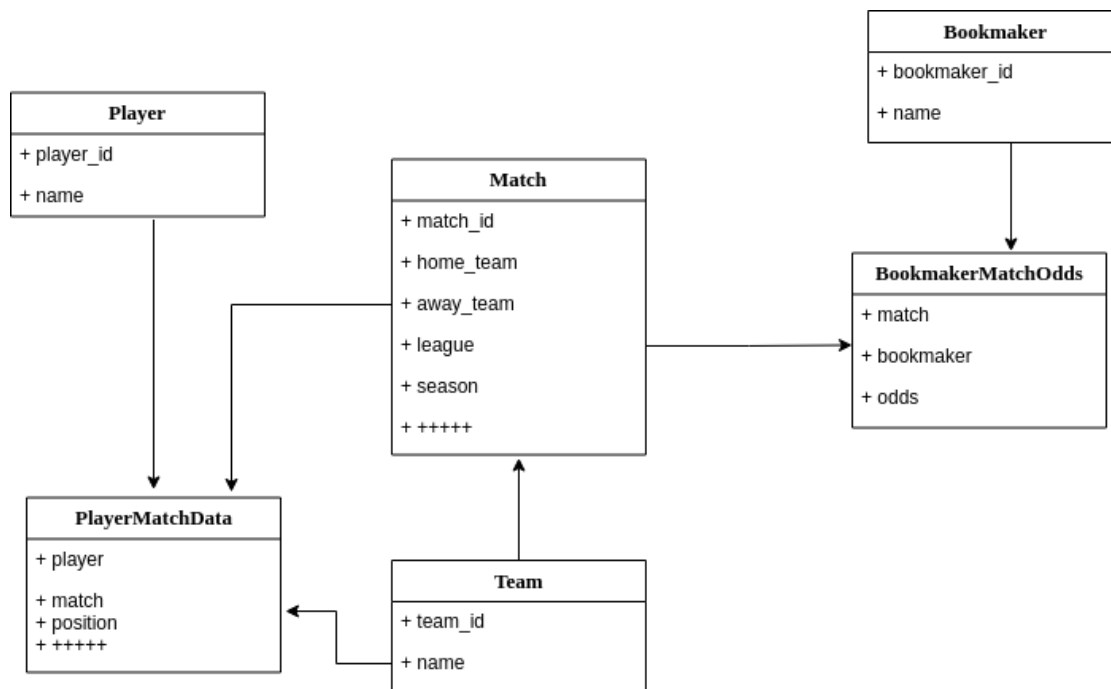


Figure 5.2: Overview of the database used by the system.

5.3.2 Neural networks

The prediction models described in Chapter 4 were implemented using the Python library Keras. To facilitate experimentation and limit duplicate code, common super-classes were created for the fully connected and sub-network models. The abstract classes act as a wrapper for Keras and handles functionality such as constructing the Keras model, training the model and saving and loading trained models. Each prediction model was implemented as a subclass of an abstract class, and handles features related to the specific model, such as which input features it needs.

5.3.3 Betting simulator

To evaluate the profitability of the prediction models, a system simulating a full season of betting was created. The simulator simulates a season of a given tournament week by week, presenting relevant data available before the game starts, along with the closing odds for each game in the week from several bookmakers.

Based on the odds and the forecasts made by the model being evaluated, bets may be placed, and their profits are recorded. The workings of the betting simulator are shown in Algorithm 2.

Algorithm 2 Betting simulator

```

 $C \leftarrow$  initial bankroll size
 $\tau \leftarrow$  confidence threshold parameter
 $\epsilon \leftarrow$  minimum probability parameter
for all match week  $\in$  season do
  bets  $\leftarrow$  empty list
  for all match  $\in$  match week do
    p  $\leftarrow$  outcome predictions from model,  $\mathbf{p} \in \mathcal{R}^3$ 
    d  $\leftarrow$  odds for the given match,  $\mathbf{d} \in \mathcal{R}^3$ 
    for all outcome  $i \in \{0, 1, 2\}$  do
      if  $d_i * p_i > 1 + \tau$  and  $p_i > \epsilon$  then
         $c_i \leftarrow$  GetBetSize( $d_i, p_i, C$ )
         $C \leftarrow C - c_i$ 
        bets  $\leftarrow$  Append( (match,  $i, d_i, c_i$ ), bets)
      end if
    end for
  end for
  for all bet ( $d_i, c_i$ )  $\in$  bets do
    if bet is successful then
       $C \leftarrow C + d_i * c_i$ 
    end if
  end for
end for

```

Different bet staking strategies are implemented by different implementations of the `GetBetSize` function, and the risk allowed in the betting procedure is controlled by the two parameters τ and ϵ .

Chapter 6

Experiments and Results

This chapter presents the experimental setup and results conducted in this thesis. The chapter first presents a description of the prediction models. A presentation of the betting simulation procedure is then presented before the experiment is conducted and its results are presented.

6.1 Experimental Plan

In our experiment, we will attempt to achieve the research goal and answer the research questions defined in Section 1.1.

The ultimate goal of the experiment is in line with the research goal of this thesis, which is to make a profit on the football betting market using data-driven machine learning methods. By training, validating and testing our suggested neural network models, which are described in Chapter 4, this experiment aims to generate a profit over two whole seasons in the English Premier League.

The performances of the two different network architectures, described in Section 4.3, will be compared and evaluated in order to investigate how the architecture affects the models' performance. The profitability, validation accuracy and model prediction quality will be assessed when comparing these architectures and by presenting and comparing these our experiment aims to discover results which can lead us to a definite answer to this particular research question.

6.2 Experimental Setup

6.2.1 Network setup

All of our suggested models mentioned in Section 4 are neural networks, which yields a large number of possible configurations. The networks could vary in loss function, optimization algorithm, network dimensions, network layer sizes and activation function.

To prevent this experiment from training several thousand models, fixed loss functions and optimization algorithms have been applied to the models. The loss function used is *categorical cross entropy* and the optimization algorithm is *Adam*.

A neural network instance can differ vastly in both size and shape. It can have a varied number of both hidden layers and the number of neurons in each layer. For our experiment, a fixed number of sizes and layers have been defined. A list of layers, layer sizes and hidden layer activation functions are iterated through, and for each configuration, five instances of the given configuration are trained. At each training instance, the training data is also shuffled, and by combining shuffled training data with multiple configuration instances, the results present the overall accuracy of that given network, and not just the effects of a potentially lucky combination of initial weights, biases and training data.

The following parameters were applied to our experiment:

- *Layer size*. The number of neurons each hidden layer contains.
 - Values: [8, 16, 32, 64].
- *Layer depth*. The number of hidden layers in the network.
 - Values: [1, 2, 3].
- *Activation function*. Which activation function the hidden layers apply.
 - Values: [Sigmoid, Tanh, ReLU].
- *Number of configuration runs*. How many networks of each configuration are trained.
 - Values: 5.

By iterating through each possible configuration listed above, this yields $4 * 3 * 3 * 5 = 180$ different networks for each of the suggested models in Chapter 5. After training, the accuracy and RPS values achieved on the validation data for each configuration is stored, and the configuration with the lowest mean RPS value for each model is selected for profitability evaluation on the test set.

6.2.2 Data

In this section we will describe the data available for each model, and how this data is split up in training, validation and test data.

The training set is the data set used to train the model. The goal is for the model to see and learn patterns from this data set.

The validation set is used to evaluate the given model. After each training epoch the validation data set is fed through the model for evaluation. The model never learns from this data set and it is only used as an unbiased evaluation of how well the model behaves with unseen data. Models which perform well on the validation data have not overfitted on the training data, and the best performing models on the validation data can be considered as the best models dealing with unseen data and is thus chosen to be tested on the test data.

The system randomly scrambles the training and validation data before splitting the data in a fixed ratio. 10% of the available training data is split into validation data and used for evaluating the models after each training epoch.

The test data is every match from the 2016/2017 and 2017/2018 seasons of the English Premier League. This data is used to provide an unbiased evaluation of a final model fit on the training dataset, which in turn will determine how well the model deals with unseen data. This is also the data which the model will predict the final match outcomes on and attempt to make a profit using the different betting strategies described in Section 6.2.3.

Expected goals of starting lineup

For the expected goals of starting lineup-model data primarily from Understat.com are used. As described in Section 5.1.1, Understat provides statistics for six different leagues dating back to the 2014/2015-season. This means there are 9860 different matches for the model to train on. Since the experiment aims to test the

models predictive power on the 2016/2017 and 2017/2018 seasons of the English Premier League these seasons are excluded from the training set. This means that in total there is $9860 - 760 = 9100$ matches for the model to train on.

Data-driven pi-football

For the two models discussed in Section 4.1 primarily data from FiveThirtyEight are used. The data is thoroughly described in Section 5.1.2.

FiveThirtyEight provides data dating back only two full seasons, but what it lacks in number of seasons it makes up in number of leagues. Statistics for over 30 different leagues is provided in the dataset, which means that there exist over 18,000 matches with data.

The data point *match importance*, which we use to model the psychology factor from pi-football, is not provided in every single match. The leagues where this statistic is missing is thus removed from the training set. In total there are 14009 matches with match importance set and by excluding the matches with insufficient data from the test set there are a total of $14009 - 760 = 13249$ matches in the training set.

Cold-start

Since all of our prediction models try to model a team's strength by using the team's performance relative to the current season in some kind of way, certain states from the data set need to be handled. To battle null-values at the start of the season which may ruin the model's ability to distinguish weak teams from teams that have yet to play in the current season, the first two rounds in each training and test season are scrapped from the data set.

Normalizing features

All data fed into a neural network is of a numeric value. These values are propagated through the network and combined with its corresponding weights is responsible for the final output for the network. Since each input is multiplied with its connected weights, this results in input values with large magnitudes having a far greater influence on the final output. For example, let's say we have a neural

network with two different features, X and Y. The X feature has a value of 0.5 and the Y feature a value of 99. A neural network weight change of 0.1 will change the magnitude of the X factor by $0.1 * 0.5 = 0.05$, but the Y factor's magnitude changes by $0.1 * 99 = 9.9$. In essence, normalization is needed because it removes any biases large data points may provide in the network. If all of the data is normalized and scaled to the same range then this issue is taken care of. The normalization technique applied is *min max normalization*. See Equation 6.1 to a detailed overview of the formula.

$$I_{c,j} = \frac{f_{c,j} - f_{j,min}}{f_{j,max} - f_{j,min}} \quad (6.1)$$

Feature j for case c , i.e. $f_{c,j}$, is scaled to $I_{c,j}$ (the value of input neuron j for case c) as described in Equation 6.1 and this ensures values in the range $[0, 1]$

6.2.3 Betting evaluation

The betting performance of each model is evaluated using 5 instances of the network configuration with the lowest RPS score on the validation set. Each model is evaluated by its profit by simulating the model over the course of two seasons in the English Premier League using the betting simulation described in Section 5.3.3, with betting parameters $\tau = 0.1$ and $\epsilon = 0.1$. For each model, four of the betting strategies described in Section 3.2 are applied:

- Fixed bet, with $c_i = 1$
- Fixed return, with $c_i = \frac{1}{d_i}$
- Kelly ratio, with $C_0 = 0.05$
- Variance-adjusted

The initial bankroll is set to 100 for all strategies. The profitability of a model-strategy combination is evaluated by its Return on Investment (ROI), defined as

$$ROI = \frac{\text{End bankroll} - \text{Initial bankroll}}{\text{Cost of Investments}} \quad (6.2)$$

This profitability evaluation compared to a plain net profit return, is more fair. This is due to the fact that the Kelly ratio strategy will place bets with much

larger sizes than the other strategies since this strategy factors in the bankroll when determining bet sizes. If presenting just a plain net profit for each strategy, then the Kelly ratio strategy when successful would return way larger profits than the other strategies, and vice versa if unsuccessful. An evaluation criteria which factors in the cost of investments, i.e. how much each strategy has staked, is thus a much fairer type of evaluation.

For each model-strategy combination, the weekly development of the ROI over the span of a season is presented for the most and least profitable instances, along with the mean of all instances.

6.3 Experimental Results

This section presents the experimental results achieved in this report. For each model, the validation results and model prediction quality are presented. The profitability evaluation for both test seasons is also presented, as well as betting distributions for both seasons. The goal of this section is not to take a deep-dive into how and why each model performed the way it did, but rather give an overview of how each model fared against the test data, so that there is room for discussion and evaluation in a later section.

6.3.1 Baselines

In order to assess the networks chosen for evaluation some baseline values are needed. The baselines represent the bare minimum of what the networks should be able to achieve.

The bookmakers set their odds based on their own predictions of the match outcome. For each match, the team with the lowest odds is the team which the bookmakers deems most probable to win. A benchmark could thus be beating the profit generated by betting on the favorite for each match. If the models beat this baseline, we can safely conclude that our system has learned to predict something useful. By deeming the outcome with the lowest odds for each match as a feasible bet and applying a fixed bet-strategy, a return on investment can be generated and used as a benchmark. Betting on the least favorite in each match is also added as a benchmark in order to assess the opposite effect compared to the most likely outcome. Another benchmark is based on the home ground advantage. 46.2% of

all matches in the Premier League in the seasons from 1992/1993 to 2014/2015 ended in a home victory (SMarkets [2019]) and by applying the fixed bet strategy on a home victory for each match another baseline to beat is generated. If our models can generate a higher profit than this benchmark it shows that the networks have learned something else than to just bet on the home team.

Table 6.1 shows the return on investment for the three different benchmarks listed above.

Season	Always home	Most likely	Least likely
2016/2017	0.0710	0.0968	-0.2018
2017/2018	0.0230	-0.0378	-0.0630

Table 6.1: Return on investment for three benchmarks using the fixed bet strategy

When evaluating if the models are adequately predicting the match outcomes the fact that that the bookmakers adds a margin (described in Section 2.4.1) to each match must be taken into account. In essence this means that losing less than a given margin for each season could be considered a decent result. See Table 6.2 for the calculated margin for William Hill during the two test seasons.

Season	Margin
2016/2017	0.0470
2017/2018	0.0485

Table 6.2: Calculated margin for William Hill in the English Premier League

By inspecting the benchmarks generated in Table 6.1, it is evident that the always home-benchmark returns a ROI which exceed the margin that William Hill applied on the two seasons. The always home benchmark returns a ROI of 7.1% for the first season and a ROI of 2.3% in the second season. The most likely-benchmark returns a ROI of 9.7% after the 2016/2017 season, but fails to replicate these results in the second season, generating a negative ROI of -3.8%.

6.3.2 Fully connected expected goals of starting lineup

Table 6.3 shows the RPS values and accuracy achieved by the expected goals of starting lineup-model with a fully connected neural network. This model is thoroughly described in Section 4.2. A single hidden layer with 32 neurons and

the Rectified Linear Unit (ReLU) as the activation function generated the best validation results, and will therefore be used to evaluate the profitability of the model.

Layer sizes	Activation	Accuracy	RPS
[8]	ReLU	0.523097826	0.20187376
[8,8]	ReLU	0.524456522	0.20287714
[8,8,8]	Tanh	0.517663043	0.20310548
[16]	ReLU	0.539402174	0.20181155
[16,16]	Sigmoid	0.517663043	0.20289664
[16,16,16]	Sigmoid	0.517663043	0.20332393
[32]	ReLU	0.520380435	0.20153609
[32,32]	Sigmoid	0.514945652	0.20286488
[32,32,32]	Tanh	0.516304348	0.20296864
[64]	ReLU	0.529891304	0.20233367
[64,64]	Tanh	0.531250000	0.20224941
[64,64,64]	Tanh	0.527173913	0.20185789

Table 6.3: Fully connected expected goals of starting lineup - validation results

Model prediction quality

This section presents the model's predictions for all matches in the test set compared to its corresponding match outcome. The section will visualize how the model fares in its prediction compared the actual outcomes of each match, and figures will be presented to visualize these correlations. This is a qualitative analysis meant to provide insight into how the model's prediction quality is calibrated across the outcome distributions.

Figure 6.1 illustrates all the predicted probabilities for the model relative to the actual outcomes. Each prediction receives a y-label based on the actual outcome, 1 if the predicted outcome occurred and 0 otherwise. A cubic smoothing spline is then applied to the prediction-outcome points to approximate the distribution of outcomes relative to predictions. The goal by illustrating this is to explore the quality of the predicted probabilities from the model compared to the actual outcomes. A dotted line is plotted to visualize the ideal distribution of outcomes and probabilities, where probability equals outcome distribution at each point, $x = y$. This line represents the baseline for which the predictions from the model should follow. If the model's prediction is in close proximity to this line, the closer the model's predictions are to the true probabilities for each match.

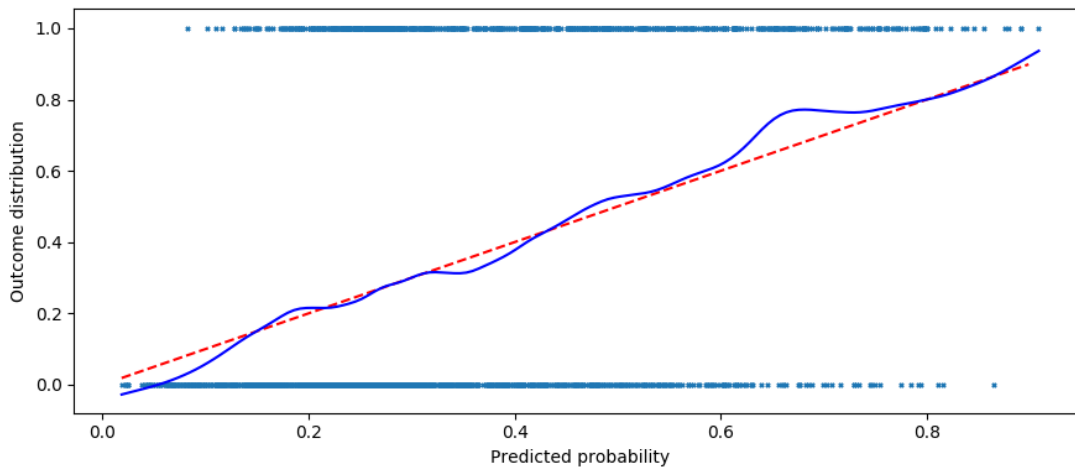


Figure 6.1: Predicted probabilities compared to actual outcomes, fully connected expected goals of starting lineup.

The smoothed distribution-curve of probabilities, represented in Figure 6.1, shows that the model is relatively decent in predicting the probabilities of match outcomes. Bar some minor fluctuations the model follows the ideal red-dotted line rather closely. A little bump can be spotted in the $[0.65, 0.75]$ region of the x-axis. At the 0.7 mark, the model expects to be correct 70% of the time. By the blue curve, we see that the actual distribution of these outcomes is around 80%, meaning that the model underestimates the probabilities in this region. In this region, the model is at risk of missing out of some bets, since the model predict the outcomes with a 70% probability, while the actual probability is 80%. The model also struggles with a slight dip in the $[0, 0.1]$ interval, and in this region the model has a tendency to overestimate the outcomes. This might lead the model to being too eager when placing bets, should the odds correspond to the ideal line. This is not an issue in our betting simulation though, since the minimum probability parameter for our betting simulator, described in Section 5.3.3, is set to 0.1. The model is relatively well calibrated along the ideal line, but the smoothed distribution is not enough to guarantee a profit, though it displays the inherent potential of the model.

Betting results

Figure 6.2 and 6.3 shows the development of the ROI for all of the betting strategies generated by the model across the 2016/2017 and 2017/2018 seasons in the English Premier League.

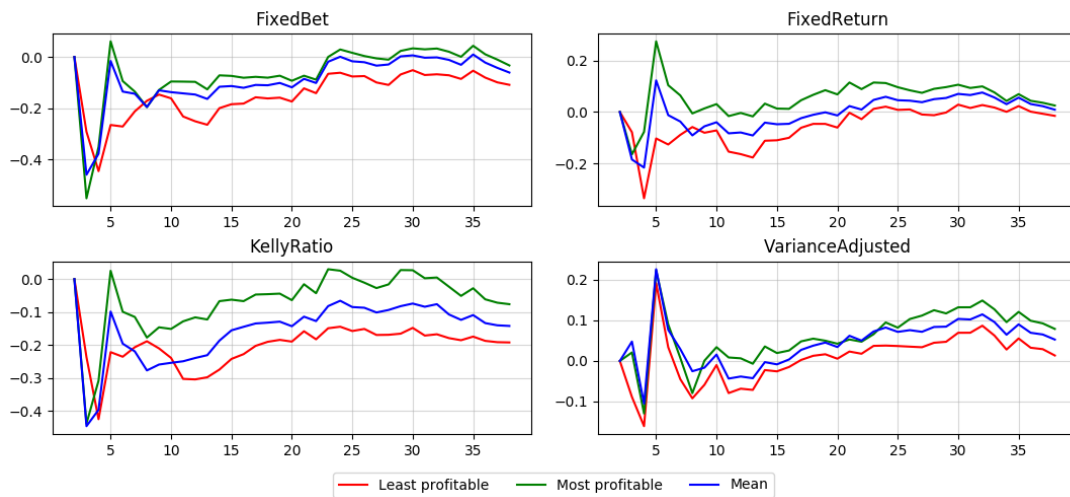


Figure 6.2: ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using the fully connected expected goals of starting lineup model.

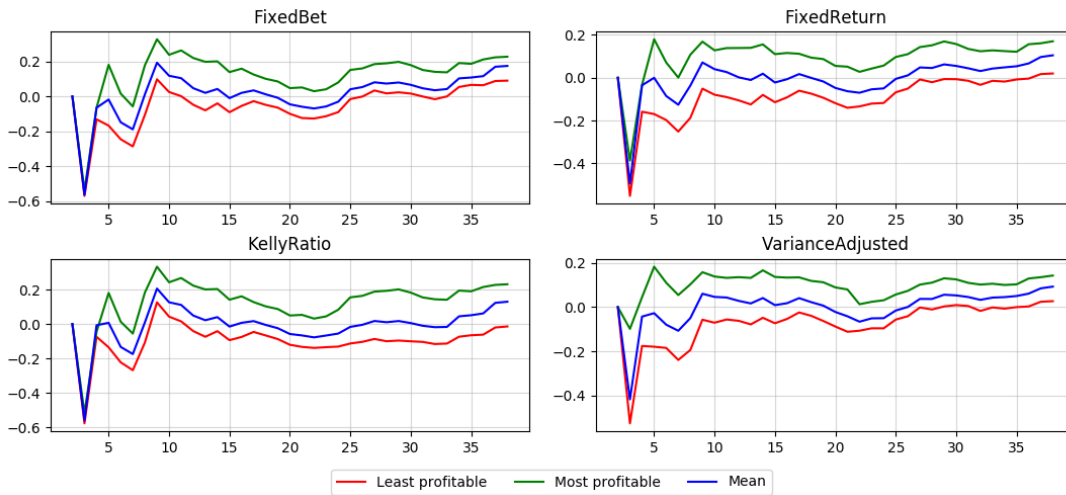


Figure 6.3: ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using the fully connected expected goals of starting lineup model.

Table 6.4 shows the generated return on investment for the model at the end of the 2016/2017 and 2017/2018-season of the English Premier League. The table shows the most profitable, least profitable and mean profitable return on investment for each of the betting strategies across the five simulations for both seasons.

Strategy	2016/2017			2017/2018		
	Min	Max	Mean	Min	Max	Mean
Fixed bet	-0.1100	-0.0332	-0.0610	0.0905	0.2277	0.1750
Fixed return	-0.0150	0.0252	0.0089	0.0200	0.1702	0.1050
Kelly Ratio	-0.1921	-0.0761	-0.1421	-0.0141	0.2311	0.1298
Variance-adjusted	0.0134	0.0789	0.0527	0.0270	0.1428	0.0928

Table 6.4: ROI after the 2016/2017 and 2017/2018 seasons for all betting strategies for the fully connected expected goals of starting lineup-model.

As seen from the table, there are two strategies which return a profit after the first season is finished. The variance-adjusted and fixed return strategies generates a ROI of 5.3% and 0.9% respectively. During the latter season, all of the strategies return a profit at the end of the season. The fixed bet strategy is the most profitable strategy this season, generating a mean ROI of 17.5%. Each strategy this season generated a ROI over 9%. The table also shows that the variance-adjusted strategy

returns a positive ROI for all of the model instances for both of the seasons.

Figure 6.4 shows the bet distribution for an instance of the fully connected expected goals of starting lineup model during the 2016/2017 and 2017/2018 seasons.

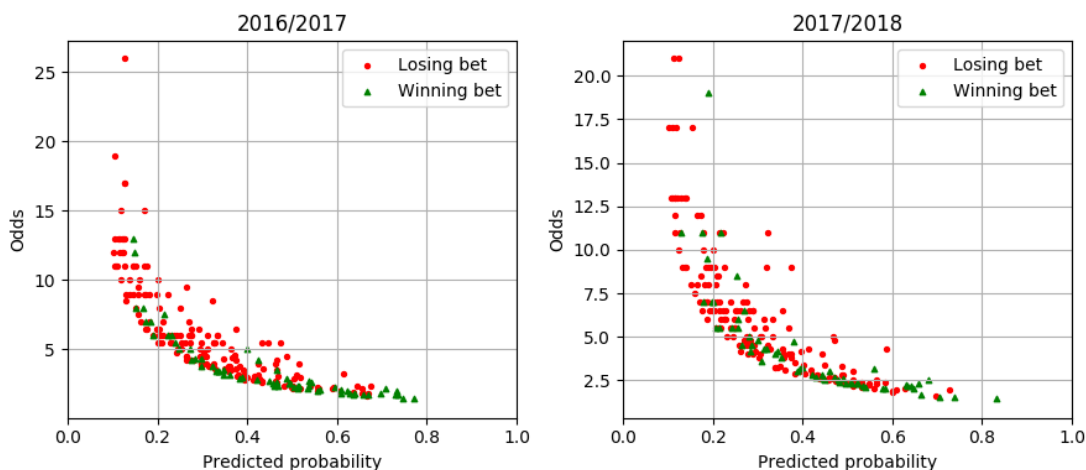


Figure 6.4: Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.

Model summary

This model manages to achieve consistently good results over the course of the two seasons for two different betting strategies. These two strategies are the fixed return and the variance-adjusted strategies. The fixed bet strategy generates the highest ROI of the second season, but is barely beneath the profit line with a ROI of -0.6% for the first season, while the Kelly ratio strategy is hugely unsuccessful in the first season, returning a ROI of -14%.

In Section 6.3.1 the baselines for our experiment are presented. The calculated margins that William Hill adds are 4.7% and 4.8% for the respective seasons, and thus any strategy which exceeds $ROI = 1 - margin$ for each season can be considered a decent result. Each strategy except the Kelly ratio strategy is successful in beating William Hill's margin. Looking at the betting-baselines, the always home-strategy was the most successful baseline with a ROI of 7.1% and 2.3%. None of the strategies manage to beat this baseline for the first season, but

all of them are able to outperform the baseline in the second season.

Another interesting observation from Figure 6.4 is the fact that there are numerous bets that exceeds the bookmakers' implied probabilities. The model tends to overestimate the probability for certain match outcomes it deems feasible. By inspecting the figure, we can see that the lower threshold for placing bets is along the curve $probability * odds = 1.1$, as specified by the betting simulator parameters. The further away bets are from this curve, the more confident is our model on that match outcome relative to the odds. These bets have a tendency to fail for this given model while the bets closer to the curve has a higher success rate. This is not entirely surprising as the bookmakers can be considered very decent in predicting match outcomes. Model predictions where $p * w \gg 1$ are more likely to be caused by the model's prediction being off, rather than the bookmaker.

6.3.3 Expected goals of starting lineup with sub-networks

Table 6.5 shows the RPS values and accuracy which the expected goals of starting lineup-model grouped in sub-networks generated. This model is thoroughly described in Section 4.2. A single hidden layer with 32 neurons and the hyperbolic tangent function (tanh) as the activation function generated the best validation results, and will thus be used to evaluate the profitability of the model.

Layer sizes	Activation	Accuracy	RPS
[8]	Tanh	0.464673913	0.20992414
[8,8]	Tanh	0.471467391	0.210191052
[8,8,8]	Tanh	0.46875	0.210135152
[16]	Tanh	0.463315217	0.210083199
[16,16]	Tanh	0.46875	0.209987271
[16,16,16]	Tanh	0.472826087	0.210410086
[32]	Tanh	0.472826087	0.209659552
[32,32]	Sigmoid	0.471467391	0.210132925
[32,32,32]	Sigmoid	0.466032609	0.210163158
[64]	Tanh	0.476902174	0.209862298
[64,64]	Sigmoid	0.471467391	0.210077992
[64,64,64]	Sigmoid	0.471467391	0.210124469

Table 6.5: Expected goals of starting lineup with sub-networks - validation results

Model prediction quality

Figure 6.13 illustrates all the model's predicted probabilities relative to the actual outcomes. The dotted line represents the baseline for which the predictions from the model should follow, while the blue curve is the approximation of the distribution of outcomes relative to the model predictions.

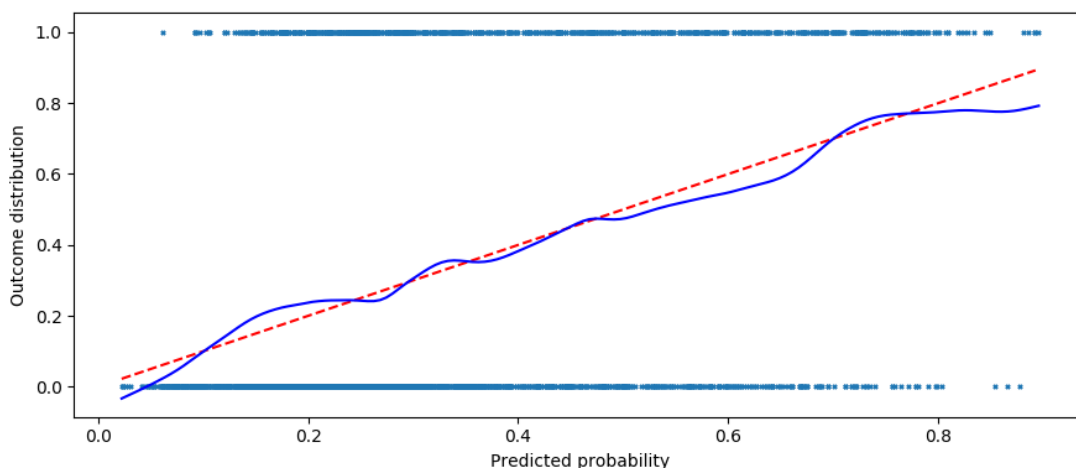


Figure 6.5: Predicted probabilities compared to actual outcomes, expected goals of starting lineup with sub-networks.

By inspecting the smoothed distribution of probabilities relative to the outcomes, several minor fluctuations can be spotted. The model struggles to follow the ideal red-dotted line for two different intervals, and the most notable ones are $[0.1,0.2]$ and $[0.5,0.7]$. For the first interval, the model expects to be correct 20% of the time, but by inspecting the blue curve one observes that the actual distribution of these outcomes is around 30%. This means that the model underestimates the probabilities in this region, and this can lead to the model being too conservative when deciding to bet on outcomes, should the odds correspond to the ideal line in this region.

For the other interval, the model has a tendency to overestimates the probabilities. Across the $[0.5,0.7]$ interval, the model consistently finds itself under the ideal line. This may lead the model to being too eager when placing bets, should the odds correspond to the ideal line in this region. Compared to its fully connected version, the sub-network model looks qualitatively weaker in terms of prediction quality, and thus weaker betting results is to be expected.

Betting results

Figure 6.6 and 6.7 shows the development of the ROI for all of the betting strategies generated by the model across the 2016/2017 and 2017/2108 season in the English Premier League.

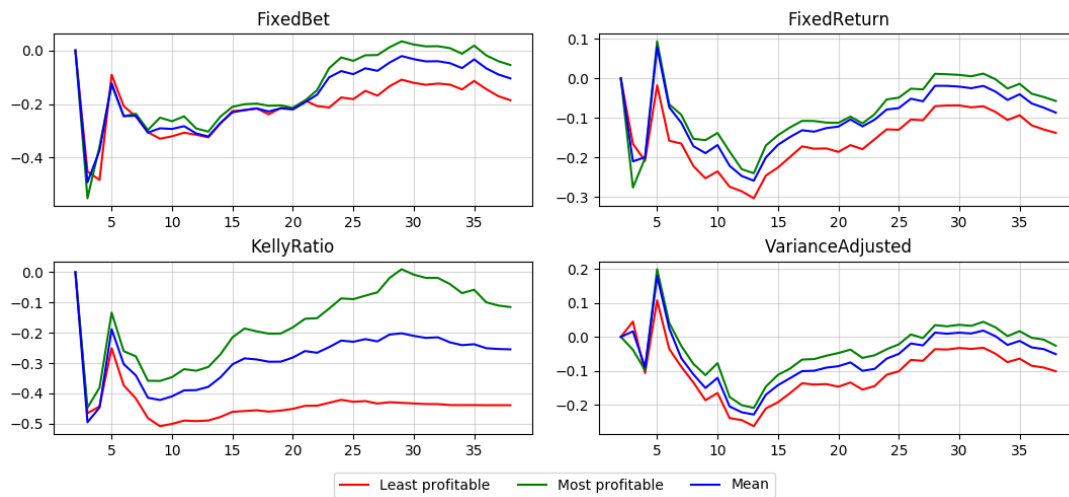


Figure 6.6: ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using the expected goals of starting lineup model with sub-networks.

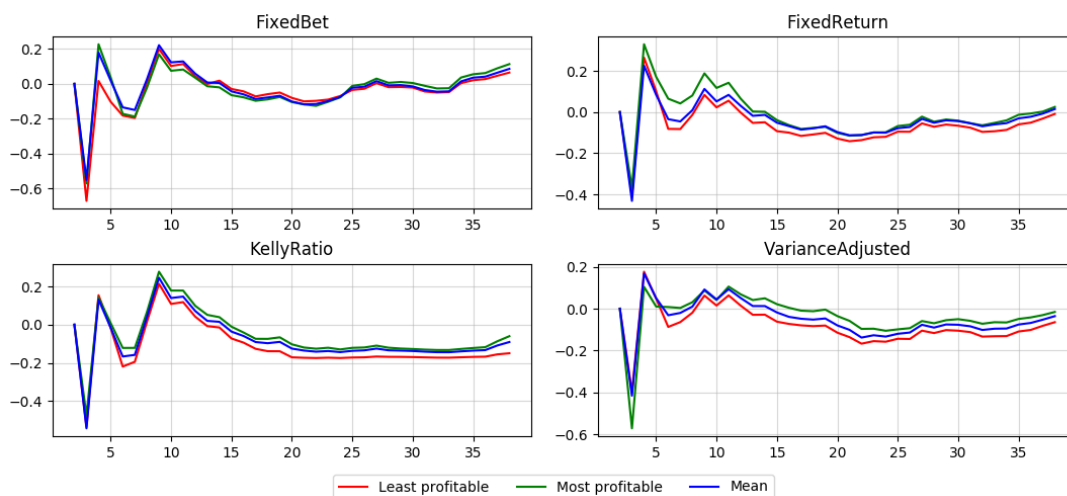


Figure 6.7: ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using the expected goals of starting lineup model with sub-network.

Table 6.6 shows the generated return on investment for the model at the end of the 2016/2017 and 2017/2018-season of the English Premier League. The table shows the most profitable, least profitable and mean profitable return on investment for each of the betting strategies across the five simulations.

Strategy	2016/2017			2017/2018		
	Min	Max	Mean	Min	Max	Mean
Fixed bet	-0.1858	-0.0544	-0.1044	0.0645	0.1130	0.0858
Fixed return	-0.1375	-0.0570	-0.0863	-0.0092	0.0248	0.0146
Kelly Ratio	-0.4391	-0.1150	-0.2546	-0.1494	-0.0602	-0.0907
Variance-adjusted	-0.1015	-0.0262	-0.0510	-0.0644	-0.0156	-0.0356

Table 6.6: ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the expected goals of starting lineup-model with sub-networks.

As can be seen from the table, neither of the strategies manage to generate a positive ROI after the 2016/2017 season. Not a single instance of the generated models were able to generate a positive return. For the 2017/2018 season, two of the four strategies manage to generate a positive return. The fixed bet strategy is the strategy which generates the highest ROI with 8.58%.

Figure 6.8 shows the bet distribution for an instance of the expected goals of starting lineup with sub-networks model during the 2016/2017 and 2017/2018 seasons.

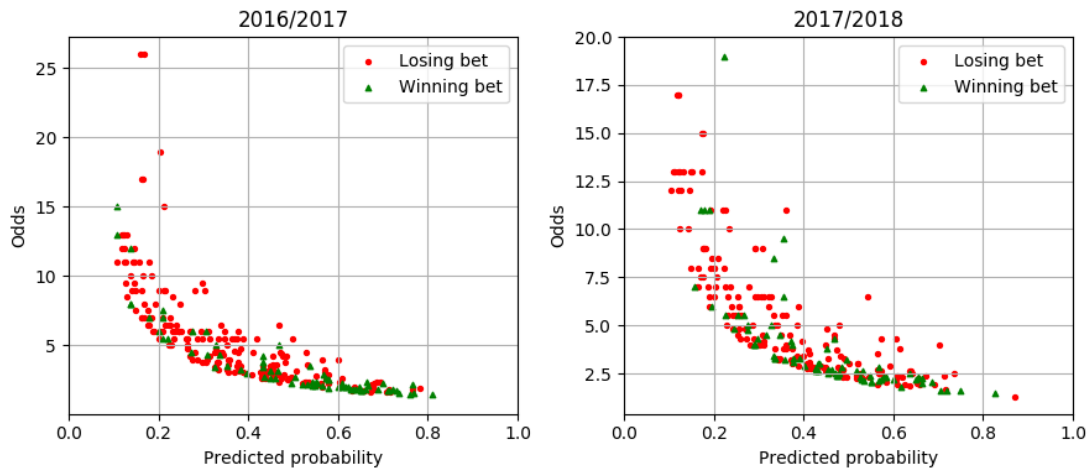


Figure 6.8: Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.

Model summary

The expected lineup-model when grouped into sub-networks fails to generate consistent results over the course of two seasons in the English Premier League. All strategies yield a negative return on investment during the first season, and thus the positive results achieved for two of the models in the second season is insignificant.

Figure 6.8 illustrates the relationship between the odds from the bookmakers and the probabilities predicted by the model for each bet the model placed, and by inspecting this figure we see a tendency of struggling with overestimating the outcomes. As the bookmakers can be considered very decent in predicting match outcomes, model predictions where $p * w \gg 1$ are more likely to be caused by the model's prediction being off, rather than the bookmaker. The sub-network model struggles with a lot of outcomes being predicted with a higher probability than the odds should imply, which in turn results in the model being unsuccessful across the two test seasons.

When inspecting Figure 6.5 and 6.1, which depicts the model prediction quality of the models, we suggested that the sub-network version would struggle more when run through the betting simulation than the fully connected version. This turned out to be very much the case, as the fully connected version managed a profit across the two seasons, while the sub-network failed to accomplish this. The sub-network model struggled to follow the optimal distribution of outcomes in its predictions and the betting results across the two test seasons is evidence of this.

6.3.4 Fully connected pi-football

Table 6.7 shows the RPS values and accuracy which the data-driven pi-football-model with a fully connected neural network generated. This model is thoroughly described in Section 4.1. A single hidden layer with 32 neurons and the Rectified Linear Unit (ReLU) as the activation function generated the best validation results, and will therefore be used to evaluate the profitability of the model.

Layer sizes	Activation	Accuracy	RPS
[8]	ReLU	0.592297477	0.180972756
[8,8]	ReLU	0.585657371	0.181313198
[8,8,8]	Tanh	0.593625498	0.181691941
[16]	ReLU	0.588313413	0.181014172
[16,16]	ReLU	0.592297477	0.181367379
[16,16,16]	ReLU	0.590969456	0.18143818
[32]	ReLU	0.586985392	0.18095835
[32,32]	ReLU	0.588313413	0.181545909
[32,32,32]	ReLU	0.594953519	0.181569471
[64]	ReLU	0.586985392	0.181361335
[64,64]	Sigmoid	0.586985392	0.181505336
[64,64,64]	Sigmoid	0.584329349	0.181534505

Table 6.7: Fully connected pi-football - validation results

Model prediction quality

Figure 6.9 illustrates all the model's predicted probabilities relative to the actual outcomes. The dotted line represents the baseline for which the predictions from the model should follow, while the blue curve is the approximation of the distribution of outcomes relative to the model predictions.

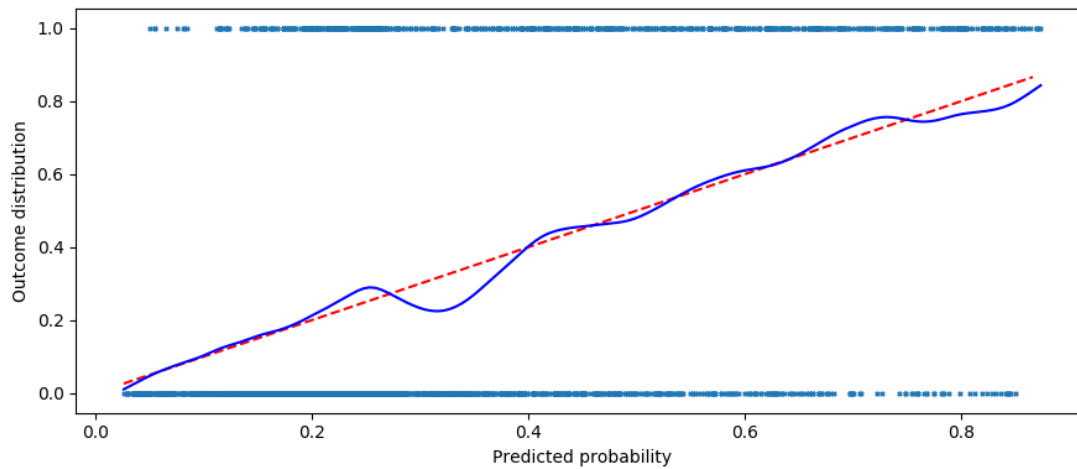


Figure 6.9: Predicted probabilities compared to actual outcomes, fully connected pi-football.

The smoothed distribution of probabilities displays that the model is relatively decent in predicting the probabilities of match outcomes. By inspecting the figure one observes that the curve follows the red-dotted ideal line rather closely, except for a little dip that can be spotted in the $[0.25, 0.35]$ region of the x-axis. At the 0.3 mark, the model expects to be correct 30% of the time. By the blue curve, we see that the actual distribution of these outcomes is around 20%, meaning that the model overestimates the probabilities in this region. This may lead the model to being too eager when placing bets, should the odds correspond to the ideal line.

Betting results

Figure 6.10 and 6.11 shows the development of the ROI for all of the betting strategies generated by the model across the 2016/2017 and 2017/2018 seasons in the English Premier League.

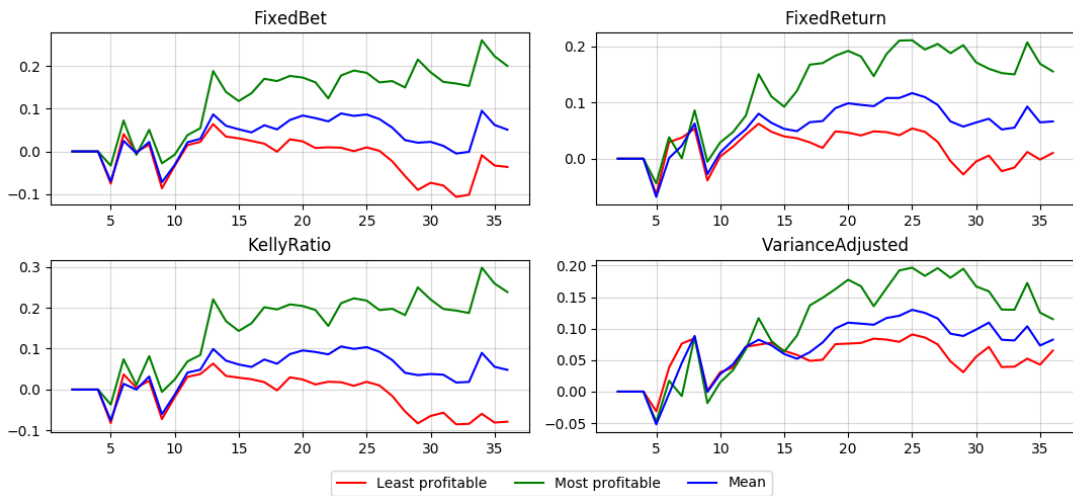


Figure 6.10: ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using fully connected data-driven pi-football model.

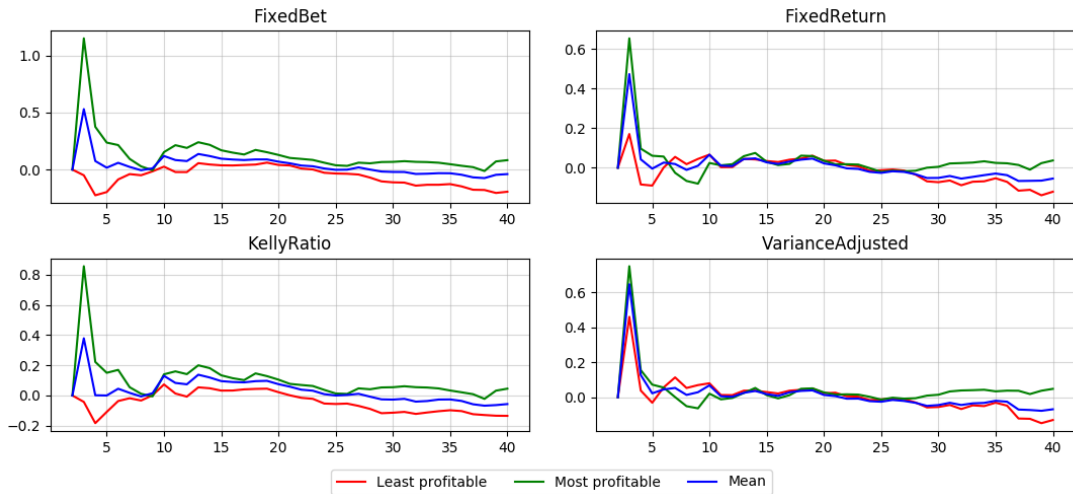


Figure 6.11: ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using fully connected data-driven pi-football model.

Table 6.8 shows the generated return on investment for the model at the end of the 2016/2017 and 2017/2018 seasons of the English Premier League. The table shows the most profitable, least profitable and mean profitable return on investment for each of the betting strategies across the five simulations.

Strategy	2016/2017			2017/2018		
	Min	Max	Mean	Min	Max	Mean
Fixed bet	-0.0364	0.2010	0.0510	-0.1939	-0.0827	-0.0394
Fixed return	0.0103	0.1557	0.0665	-0.1210	-0.0368	-0.0546
Kelly Ratio	-0.0784	0.2383	0.0481	-0.1350	0.0452	-0.0569
Variance-adjusted	0.0651	0.1151	0.0824	-0.1301	0.0483	-0.0686

Table 6.8: ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the fully connected pi-football-model.

The table shows that the model manages to generate sizeable positive results for the 2016/2017 season for every strategy, with each strategy generating a ROI of over 4%. The variance-adjusted strategy is the most profitable strategy with a ROI of 8.24%. For the second season, neither of the strategies manages to generate a positive ROI after the 2017/2018 season, the best strategy generating a ROI of -4%.

Figure 6.12 shows the bet distribution for an instance of the fully connected data-driven pi-football model during the 2016/2017 and 2017/2018 seasons.

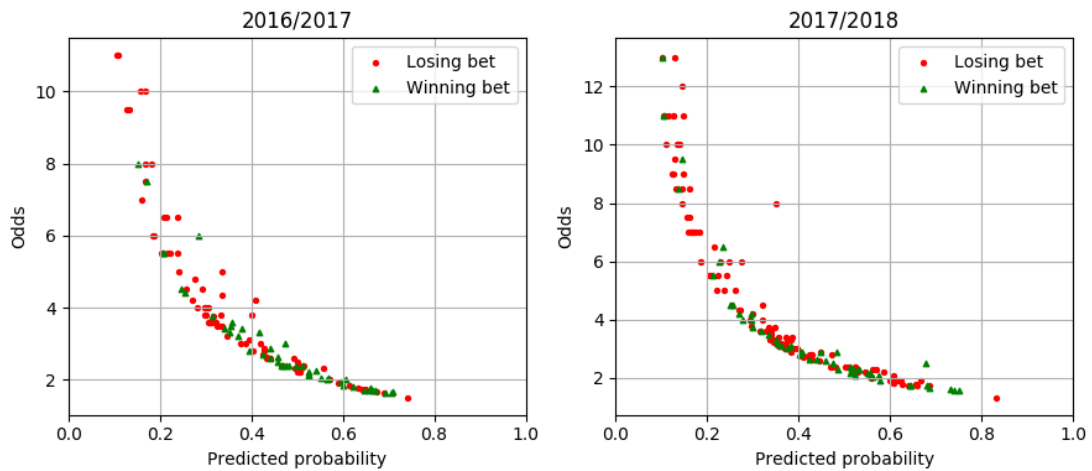


Figure 6.12: Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.

Model summary

This model failed to achieve consistent result over the course of two full seasons in the English Premier League. The first season, each strategy performed well and gained profit. For the second season, however, it failed to do so. None of the strategies managed a mean positive return on investment for the second season.

It is worth noting that some instances of the model managed to stay on the positive side for the second season, with the Kelly ratio and variance-adjusted strategies gaining a profit of 4% for their best instances. The fixed bet strategy on average also manages to beat the built-in margin by William Hill which was 4.8% this season, and can thus be considered a decent result.

When investigating Figure 6.12, one sees that most of feasible bets generated by the model is in close proximity to the curve $odds * probability \approx 1.1$, which is the lower threshold for which the betting strategies operates. The further away bets are from this curve, the more confident is our model on that match outcome relative to the odds. By staying close to this curve it shows that the model and the bookmakers are relatively close to each other when predicting matches, which speaks for the models prediction quality, as agreeing with the bookmakers is decent since they are good at predicting match outcomes. A consequence of staying too

close to the bookmaker's odds, however, is that the bookmaker's margin, described in Section 2.4.1, kicks in. Since the bookmakers add an overhead to their odds in an attempt to ensure a profit, the models will lose money if they are in line with the bookmakers prediction.

6.3.5 Data-driven pi-football with sub-networks

Table 6.9 shows the RPS values and accuracy which the data-driven pi-football-model with sub-networks generated. This model is thoroughly described in Section 4.1. A single hidden layer with 32 neurons and the Rectified Linear Unit (ReLU) as the activation function generated the best validation results, and will thus be used to evaluate the profitability of the model.

Layer sizes	Activation	Accuracy	RPS
[8]	ReLU	0.52457	0.199803
[8,8]	Tanh	0.51793	0.199910
[8,8,8]	Tanh	0.51527	0.199450
[16]	ReLU	0.50600	0.199081
[16,16]	Tanh	0.50996	0.199236
[16,16,16]	Tanh	0.51660	0.199464
[32]	ReLU	0.51527	0.198822
[32,32]	Sigmoid	0.51129	0.199460
[32,32,32]	Tanh	0.51800	0.199673
[64]	ReLU	0.52191	0.198829
[64,64]	Tanh	0.50600	0.199404
[64,64,64]	Sigmoid	0.51527	0.199011

Table 6.9: Pi-football with sub-networks - validation results

Model prediction quality

Figure 6.13 illustrates all the model's predicted probabilities relative to the actual outcomes. The dotted line represents the baseline for which the predictions from the model should follow, while the blue curve is the approximation of the distribution of outcomes relative to the model predictions.

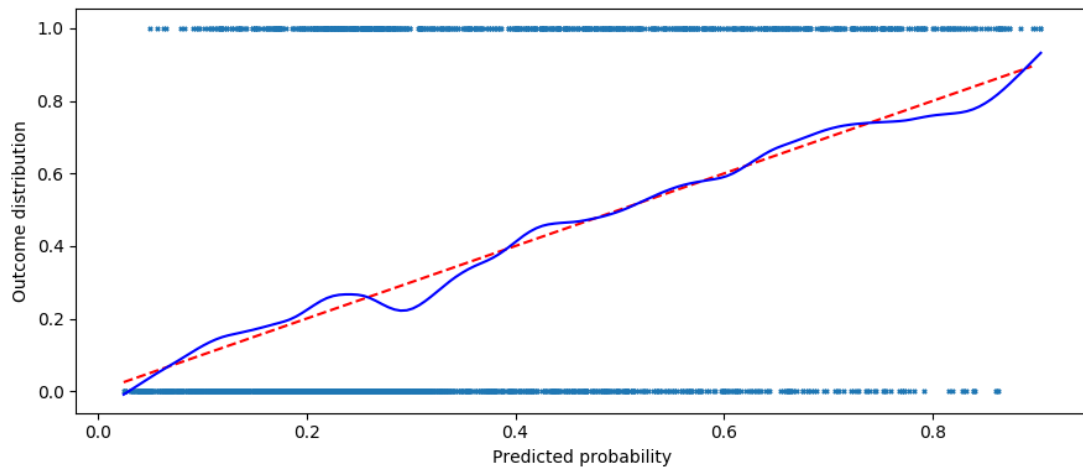


Figure 6.13: Predicted probabilities compared to actual outcomes, pi-football with sub-networks.

The smoothed distribution of probabilities displays that the model is relatively decent in predicting the probabilities of match outcomes. By inspecting the figure one observes that the curve follows the red-dotted desired line rather closely. A little dip can be spotted in the $[0.25, 0.35]$ region of the x-axis. At the 0.3 mark, the model expects to be correct 30% of the time. By the blue curve, we see that the actual distribution of these outcomes is around 20%, meaning that the model overestimates the probabilities in this region.

An interesting observation is that the fully connected pi-football model and the sub-network model are very similar in their shape, though not entirely surprising given that they share features. Both models follow the same curve for most of the intervals, only the difference being the fully connected model having slightly larger deviations from the ideal line than the sub-network version.

Betting results

Figure 6.14 and 6.15 shows the development of the ROI for all of the betting strategies generated by the model across the 2016/2017 and 2017/2018 seasons in the English Premier League.

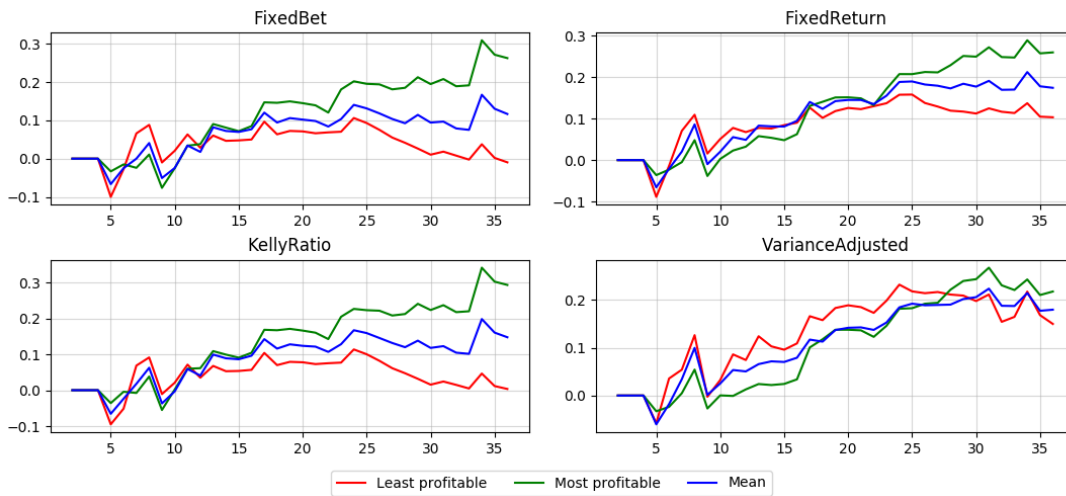


Figure 6.14: ROI for all of the betting strategies over the span of the 2016/2017 season in the English Premier League using data-driven pi-football model with sub-networks.

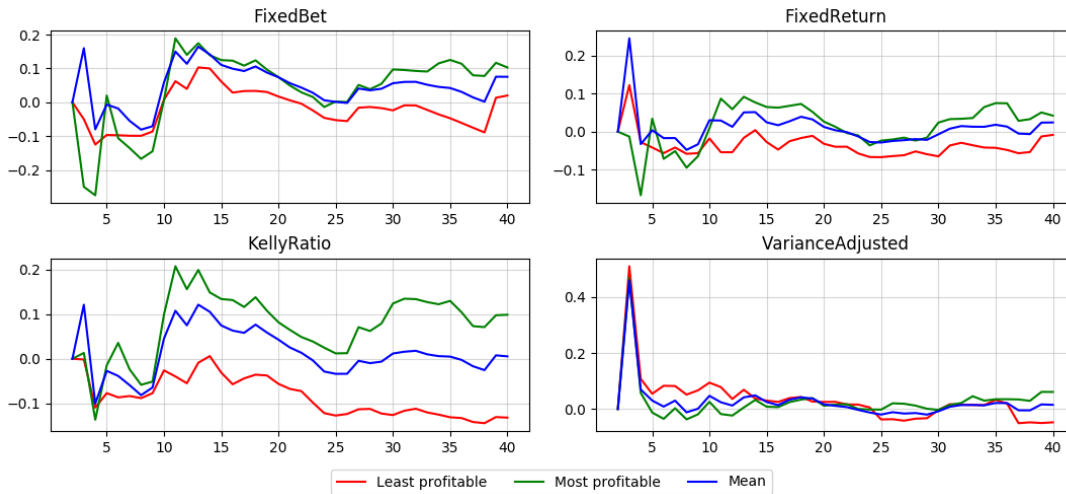


Figure 6.15: ROI for all of the betting strategies over the span of the 2017/2018 season in the English Premier League using data-driven pi-football model with sub-networks.

Table 6.10 shows the generated return on investment for the model at the end of the 2016/2017 and 2017/2018 seasons of the English Premier League. The table shows the most profitable, least profitable and mean profitable return on investment for each of the betting strategies across the five simulations.

Strategy	2016/2017			2017/2018		
	Min	Max	Mean	Min	Max	Mean
Fixed bet	-0.0096	0.2630	0.1168	0.0200	0.1030	0.0753
Fixed return	0.1034	0.2599	0.1744	-0.0085	0.0420	0.0242
Kelly Ratio	0.0034	0.2930	0.1474	-0.1318	0.0986	0.0054
Variance-adjusted	0.1500	0.2180	0.1800	-0.0480	0.0610	0.0150

Table 6.10: ROI overview after the 2016/2017 and 2017/2018 seasons for all betting strategies for the pi-football-model with sub-networks.

The table displays that each strategy on average manages to generate a positive return on investment. After the 2016/2017 season for the data-driven pi-football-model with sub-networks the variance-adjusted strategy is the most profitable with a ROI of 18%. For the latter season, each of the four strategies also manages to generate a positive return on investment. During this season the fixed bet strategy is the most profitable one with a ROI of 7.53%.

Figure 6.16 shows the bet distribution for an instance of the data-driven pi-football with sub-networks model during the 2016/2017 and 2017/2018 seasons.

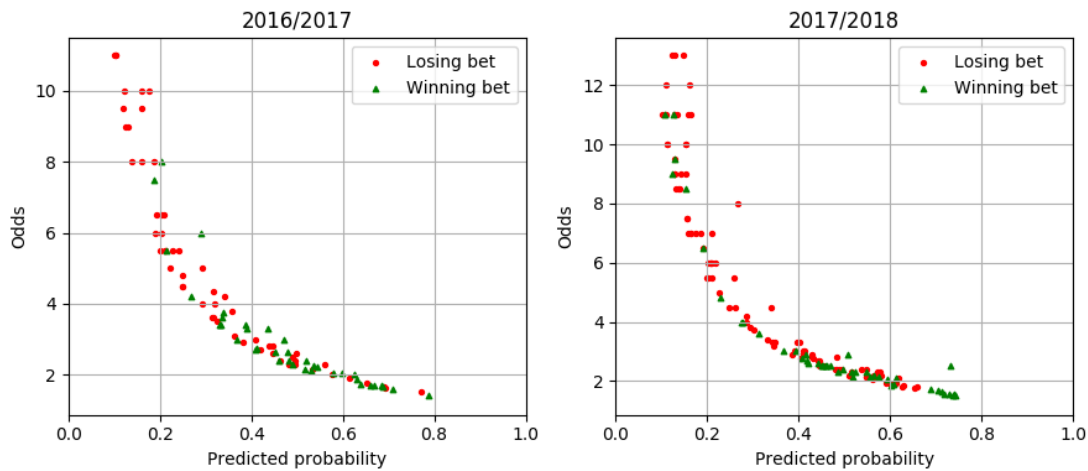


Figure 6.16: Bet distribution for the model visualized by offered odds and predicted probabilities over the span of the 2016/2017 and 2017/2018 seasons in the English Premier League.

Model summary

This model managed to achieve consistently good results for all strategies over the course of two seasons in the English Premier League. The most stable strategy across the models in terms of ROI is the fixed bet strategy, generating a ROI of 11% the first season and a ROI of 7.5% the second season, which can be considered pretty decent.

By investigating Figure 6.16 and 6.12, it is evident that the sub-network model and the fully connected model for pi-football are relatively similar. This is not surprising, as the networks share the same features, but what separates these two models is the number of matches they deem feasible. The sub-network-model is much more conservative compared to the fully connected model, placing a lower number of bets per season, and as a result is successful in generating a profit for both seasons. Neither of the models have great discrepancies when comparing the predicted probabilities with the odds, and most of the predicted match outcomes follow the curve $odds * probability \approx 1.1$.

Figure 6.13 illustrates the model's predicted probabilities relative to the actual outcomes, and this figure displays a little dip in the [0.25,0.35] region of the x-axis. The model expected to be correct around 30% of the time in this region,

but the model's curve showed that the distribution of outcomes is around 20% in this region. By comparing this figure with Figure 6.16, which is the distribution of bets which the model places, it becomes evident that the model struggles in this region. The model places 22 bets in this region across the two seasons and 5 of them are successful. This yields an accuracy of 22%, which is in line with the distribution illustrated in Figure 6.5.

By inspecting the experiment's baselines, which are described in Section 6.3.1, it is evident that several strategies beat the betting-baselines for both seasons. The always-home-baseline returns a ROI of 7.1% and 2.3% for the first and second season respectively, and both the fixed bet strategy and the fixed return strategy exceed these values for both seasons. Given that each strategy returns a profit for all strategies it also is evident that they beat the built-in margin by William Hill quite clearly.

Discussion

This chapter presents a thorough review of the results and findings from our conducted experiments. Prediction model performance comparison, betting strategy evaluation, as well as possible limitations to our experiments will be discussed and presented.

7.1 Model performance comparison

The results from Chapter 6 show that the choice of prediction model is crucial in terms of model accuracy and profitability. Tables 7.1 and 7.2 show the mean ROI achieved by each model and each strategy across the two test seasons in the English Premier League.

Model	Fixed bet	Fixed return	Kelly ratio	Variance adjusted
xG of starting lineup fully connected	-0.0610	0.0089	-0.1421	0.0527
xG of starting lineup sub-networks	-0.1044	-0.0863	-0.2546	-0.0510
Data-driven pi-football fully connected	0.0510	0.0665	0.0481	0.0824
Data-driven pi-football sub-networks	0.1168	0.1744	0.1474	0.1800

Table 7.1: Mean ROI for the models after the 2016/2017 season

Model	Fixed bet	Fixed return	Kelly ratio	Variance adjusted
xG of starting lineup fully connected	0.1750	0.1050	0.1298	0.0928
xG of starting lineup sub-networks	0.0858	0.0146	-0.0907	-0.0356
Data-driven pi-football fully connected	-0.0394	-0.0546	-0.0569	-0.0686
Data-driven pi-football sub-networks	0.0753	0.0242	0.0054	0.0150

Table 7.2: Mean ROI for the models after the 2017/2018 season

Two of the models were able to generate a profit across the two seasons by applying the betting strategies, namely the fully connected expected goals of starting lineup and the data-driven pi-football with sub-networks-model. The two other models did not achieve consistent results and generated a profit for one season, while failing to do so for the other season.

Based on the profitability evaluation, the data-driven pi-football model with sub-networks is the model which generates the most profitable and consistent results. Each strategy is successful in generating a positive ROI for both of the seasons in the test set. The most consistent strategy is the fixed bet strategy, generating a ROI of 11% and 7.5%. The other strategies are also successful, but the discrepancy between the seasons in terms of ROI is greater for these strategies.

The fully connected expected goals of starting lineup is also successful generating a profit across the two test seasons, but not for every strategy. The fixed return and the variance-adjusted strategies manage to gain a profit for both of the seasons, while the fixed bet and Kelly ratio strategies fail to do so.

Based on the results conducted in our experiment and by inspecting Table 7.1 and 7.2, the data-driven pi-football-model with sub-networks applying the fixed bet strategy is the most consistent model-strategy-combination. The model manages to beat the bookmaker's built-in margin as well as returning a substantial amount of profit, generating the highest average ROI across the two seasons with 9.5%. Based on these findings, this is the recommended combination which should be used when attempting to make a profit on the betting market.

7.2 Betting strategy performance

The four betting strategies explored in this report produced varying results, each of them generating both profit and loss across the test seasons.

The Kelly ratio strategy is the most high-risk, high-reward strategy. Ranging from generating ROIs of -44% and 29% for the different model instances, this shows the strategy's high potential gain while also showing the high risk involved. This is not surprising, as this strategy has a higher ceiling when it comes to maximal bet size compared to the other strategies evaluated. Neither of the other strategies factors in the bankroll when determining bet sizes, while the Kelly ratio strategy wagers an amount proportional to the current bankroll.

The fixed return, fixed bet and variance-adjusted strategies generated more stable results. For the fixed bet strategy, winning at most 17.5% and losing at most 10.4%, the fixed return with winning at most 17.4% and losing at most 8.8%, and the variance-adjusted strategy losing no more than 6.8% and gaining at most 18%.

An interesting observation is the stability of the variance-adjusted strategy, which is the only strategy that manages to return a positive ROI for both seasons for all model instances of a configuration. Each of the model instances for the fully connected expected goals of starting lineup model managed to return a positive ROI when using the variance-adjusted strategy, while failing to do so using the other strategies. The variance-adjusted strategy aims to minimize the difference between expected profit and the variance of that profit, which in turn results in reducing the bet size for low-probability bets. If one inspects Figure 6.4, which

is the betting distribution for the model, it becomes evident that the model has placed a lot of bets on low-probability outcomes. Since the variance-adjusted strategy reduces the bet sizes for these bets might be why it fares better than the other strategies.

7.3 Effect of domain knowledge structuring

One of the research questions for this thesis was whether or not utilizing domain knowledge when structuring artificial neural networks increased their predictive performance which in turn could aid the models into generating a profit. To shed light on this questions, the suggested models were implemented with two different network structures; one fully connected structure, and one using separate sub-networks. This is thoroughly described in Section 4.3.

By the profitability results, the fully connected version of the starting lineup-model outperformed the sub-network version, while the opposite is true for the pi-football model. For both models, the lesser performing architecture failed to generate a profit with any betting strategy. Based on the profitability alone, it is hard to argue that one architecture is better than the other.

The different model's performance in terms of validation accuracy is also an interesting observation when comparing their predictive power. As seen in Table 6.3 and 6.5, which are the validation results of the two different expected goals of starting lineup models, the fully connected version outperforms its sub-network competitor quite clearly. The fully connected model achieves a validation accuracy of 52%, while the sub-network version only manages an accuracy of 47%. This is also the case for the pi-football model, where the fully connected version achieves an accuracy of 58% and the sub-network is left with an accuracy of 51%. The fully connected models generates higher accuracies on average than the sub-network instances and the domain-driven structuring of networks fails to produce satisfactory results compared to the fully connected models.

Figures 6.9 and 6.13 shows the smoothed betting distribution relative to outcomes for the fully connected and the sub-network pi-football model. These figures represent the models prediction quality and by examining these figures a minor difference in prediction quality can be spotted. Both models follows a relatively similar curve across the probability distribution, and they are also pretty decent in predicting the probabilities of match outcomes, which is illustrated in how close the smoothed curve is to the ideal line. The difference between the models lies in the

deviation from the optimal line in the fully connected version. The sub-network model lies closer to the line and fully connected suffers from larger dips in the regions where the models are not in line with the optimal distribution. This indicates that the sub-network model's prediction are of a higher quality and this statement can be backed by the fact that this model performs better in the betting simulation.

Based on the contradictory findings from our conducted experiment, where the fully connected models outperforms the sub-networks in some manner of evaluation and vice versa, it is not possible to conclude that the grouping of related features into sub-networks increases neither the predictive power nor the profitability of neural network models.

7.4 Domain knowledge limitations

In one of our research questions we asked whether or not utilizing domain knowledge when structuring artificial neural networks increased their predictive performance. We answered this by comparing the performance of our prediction models, where our models were built by both structuring related features into sub-networks and plain fully-connected networks where all features shared the same input network. When grouping related features into separate components, it is obvious that a certain amount of domain knowledge is needed. To be able to identify related features to be grouped into different components requires high domain knowledge, and while we do consider ourselves as engaged football fans who follow the sport relatively closely, we still might not be equipped with enough knowledge to separate all the given features in a satisfactory way. This is thus a possible limitation of our thesis is that the related features might not be grouped correctly.

7.5 Model selection limitations

When conducting our experiments, the best configuration for each model was chosen by calculating the RPS-values from the validation data after training and then choosing the best model based on the lowest RPS-values. This model was then run through the betting simulator and evaluated by profitability.

By inspecting the best RPS-values generated from each model, it becomes evident that there is little to no correlation between RPS and generating a profit on the

betting market. For the data-driven pi-football model, the fully connected outperforms the sub-network version quite heavily when inspecting the RPS-values. The fully connected version logs a RPS-value of 0.18, while the sub-network generates a RPS of 0.198.

When comparing these two models in terms of profitability the tables have turned, however. The sub-network-version manages to generate a profit for both test seasons, while the fully connected model fails to generate such numbers. Thus, a possible limitation for this thesis is the selection of which model configuration to run through the test set. Since there is little correlation between what is evaluated for the validation data and the test data, the models can possibly suffer from this.

A better solution could be to evaluate the training and validation data in a stricter betting fashion, and this could be achieved by running the betting simulator on the validation data with a fixed strategy and logging the best profit for each configuration. Thus, one could select models from the training data based on evaluations which are completely related to the ultimate goal of the thesis, which is generating a profit.

7.6 Money management agent

Although the money management agent did well on the stylized environment with perfect predictions and random odds, it did not converge on an useful betting strategy when using real match predictions, outcomes and odds. The idea behind using reinforcement learning for this purpose, was that an agent could learn to compensate for systemic errors in the match predictions, yielding better betting results than established strategies.

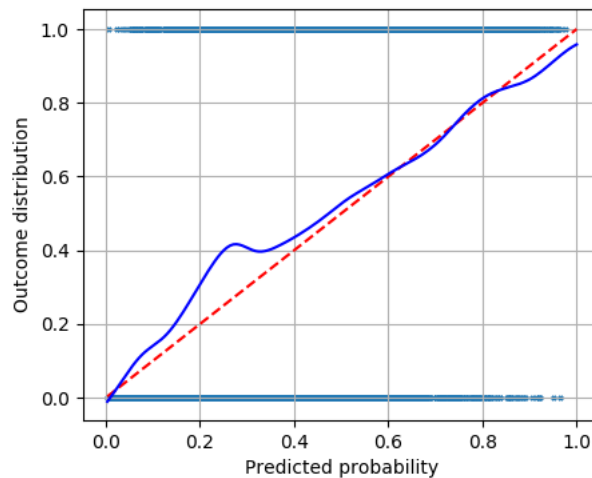


Figure 7.1: FiveThirtyEight’s predicted probabilities compared to actual outcomes

For instance, as seen in Figure 7.1, FiveThirtyEight have a tendency to underestimate the low probability outcomes, particularly around the 25% mark. If the agent had been able to learn the environment properly, it could have recognized this, and been more aggressive on the bets in this region. This might only hold up to a certain point; if the probability predictions are too far off compared to the odds and the true probability distribution, and off in a random, non-systemic fashion, there might not be enough of a pattern for the agent to recognize.

The fact that the action space was simplified significantly, only allowing the agent to place unit bets, may have hindered it from learning a profitable betting strategy. The money management strategies used in the evaluation of our prediction models (except the fixed bet strategy) modify the bet size to optimize some utility function. For example, the variance-adjusted strategy from Rue and Salvesen [2000] places bets to maximize the expected profit minus the variance of the profit, which may give more stable results. A mechanism like this is unavailable to an agent operating in our simplified fixed bet action space, giving the agent no room to mitigate risk other than refraining from betting.

Another hindrance is that the money management problem may not fit the model of a Markov Decision Process very well. In a Markov Decision Process, the state of the system should have the *Markov property*, which states that the future is independent of the past given the present. While that technically holds in our model of the betting environment, most of the state space is also wholly independent of

the *present*. The odds and probabilities observed in S_t is in no way affected by the observation S_{t-1} , and the only part of the state affected by the agent's action is the current bankroll. On the other hand, the agent operating in the stylized environment did manage to learn and improve, so the loose fit of the MDP model may not be a big issue in itself.

In summary, the results on the highly stylized environment showed promise, being able to recognize betting situations where a unit bet would be a rational decision. When applied to real odds and predicted probabilities, the agent was not able to learn a strategy that can outperform established betting strategies, most likely due to a too constrained action space not giving the agent enough options to mitigate risk.

Conclusion and Future Work

This chapter presents the conclusions drawn based on the results achieved in this report in light of the research questions. Suggestions for improvements to the system and experiments are also presented.

8.1 Conclusion

Section 1.1 presented the goal of this thesis, as well as three research questions to guide the research towards that goal. The goal of this thesis was to generate a profit on the football betting market using data-driven machine learning methods.

The first research question consisted of whether or not football matches could be adequately predicted by artificial neural networks. Chapter 4 presents two different match outcome prediction models using neural networks, the expected goals of starting lineup and the data-driven pi-football models. Chapter 6 showed that a neural network instance was able to predict the outcomes of matches with up to 58% accuracy, and this predictive performance assisted it in returning a profit on the betting market. This result displayed that football matches can in fact be adequately predicted by artificial neural networks.

The second research question asked if utilizing domain knowledge in the structure of artificial neural networks increases their predictive performance. In Section 4.3 we described how we aimed to answer this research question by grouping related features in each model into separate components in a neural network before

merging them later on in the network structure. In Chapter 6 we compared the performances of these networks compared to fully connected versions. In Section 7.3 it was concluded that the contradictory findings from the conducted experiment, where fully connected models outperforms the sub-networks in some matter of evaluation and vice versa, made it impossible to conclude that utilizing domain knowledge in the structure of neural networks increased their predictive performance.

Research question 3 concerned using reinforcement learning techniques to learn profitable money management strategies for sports betting. Our exploratory experiments in Section 4.4 showed an agent trained in a stylized environment converging on the optimal policy, but failed when operating with the uncertainty of actual match predictions and odds. This failure seems to stem from the simplification of the action space in our experiments, where the agent was not able to choose how much to wager on an outcome, leaving very little way for the agent to control the risk of the investments.

By writing a specialization report in the fall of 2018, which investigated the state-of-the-art in predicting the outcomes of football matches and found new statistical features not utilized in previous literature, several new prediction models were suggested. In this thesis the models were created, and put to the test. By conducting experiments which simulated these prediction models across two different seasons of the English Premier League, two of our prediction models managed to retain a profit for both seasons. The research goal can thus be considered fulfilled and deemed successful.

8.2 Future Work

8.2.1 Threshold for expected gain

One interesting finding from our conducted experiment was the prediction models' tendencies to overestimate the probability of match outcomes relative to the odds. This yielded in $p * w \gg 1$ for a number of matches, and by inspecting these matches the models struggled with most of these outcomes resulting in losing bets.

As the bookmakers can be considered pretty decent in predicting match outcomes, model predictions where $p * w \gg 1$ are more likely to be caused by the model's prediction being off, rather than the bookmaker. To combat this, a possible im-

provement could be to discard bets where $p * w$ is outside a specified range, for instance only placing a bet if $1.1 < p * w < 1.2$.

8.2.2 Betting-based validation criteria

As mentioned in Section 7.5, a limitation of the system is the way it chooses models to run through the betting simulation. Since there is little correlation between the evaluation of validation data and evaluation of test data, the models chosen based on the validation evaluation may not be the models which are best equipped for returning a profit. A better way could have been to formulate a betting-based validation criteria, which would more directly optimize for the end goal of making a profitable betting system. For instance, a fixed bet strategy simulation on the validation data could be a better evaluator for the prediction system.

8.2.3 Considering multiple bookmakers

In our experiments, only the odds from William Hill were considered, which is one of the largest betting companies in the world. William Hill can thus be considered very decent in setting the correct odds for outcomes which ensures a profit for the bookmakers, but as there are over 50 bookmakers who offers odds on the final outcome of English Premier League matches, a possible improvement of the system might be to consider odds from multiple bookmakers.

By looking at the odds offered by all available bookmakers and collecting the highest odds on each outcome, opportunities that generate more profit may arise, since the betting biases added by the bookmakers will be decrease by considering multiple bookmakers odds.

8.2.4 More complex action space for betting agent

To successfully and consistently profit from sports betting, just knowing whether to bet on a match or not is generally not enough. By also choosing how much to wager, a punter can attempt to account for the uncertainty and randomness of outcomes by valuing bets differently. The agents trained in our reinforcement learning experiments were constrained to only placing a unit bet on a single outcome, or to not bet at all, removing an entire dimension of control over its profits.

An obvious extension to our model of the environment would be to introduce a more complex action space, where the agent also has control of the size of the bets it is placing. One suitable option would be a subspace of \mathbb{R}^3 , with one dimension with domain $[0, b]$ for each outcome and current bankroll b . An action could then be chosen by selecting the dimension with the highest value, or alternatively allow the agent to bet on multiple outcomes simultaneously. Another option would be a discrete-continuous hybrid action space, with the discrete actions being the possible outcomes plus the no-bet action, and the wager being a continuous parameter of each discrete action. In the case of the hybrid action space, Xiong et al. [2018] propose an interesting extension to the DQN algorithm operating on hybrid action spaces without using approximation or relaxation of the action space, which could be a perfect fit for this model.

Appendix **A**

Database tables overview

Team	
Attribute	Type
Team id	int
Name	String

Player	
Attribute	Type
Player id	int
Name	String

Bookmaker	
Attribute	Type
Bookmaker id	int
Name	String

Match	
Attribute	Type
Match id	Int
Date	Date
Home team id	Int
Away team id	Int
League	String
Season	Int
Home goals	Int
Away goals	Int
Home xG	Float
Away xG	Float
Home shots	Int
Away shots	Int
Home shots on target	Int
Home team importance	Float
Away team importance	Float
Home soccerpowerindex	Float
Away soccerpowerindex	Float

BookmakerMatchOdds	
Attribute	Type
Match id	Int
Bookmaker id	Int
Home odds	Float
Draw odds	Float
Away odds	Float

BookmakerMatchOdds	
Attribute	Type
Match id	int
Bookmaker id	int
Home odds	Float
Draw odds	Float
Away odds	Float

Bibliography

- Borøy-Johnsen, S. (2017). Beating the bookmakers - Using artificial neural networks to profit from football betting. Master's thesis, NTNU.
- Constantinou, A. and Fenton, N. (2013). Profiting from arbitrage and odds biases of the european football gambling market. *Journal of Gambling Business and Economics*.
- Constantinou, A. C. and Fenton, N. E. (2012). Solving the problem of inadequate scoring rules for assessing probabilistic football forecast models. *Journal of Quantitative Analysis in Sports*, 8(1).
- Constantinou, A. C., Fenton, N. E., and Neil, M. (2012). pi-football: A Bayesian network model for forecasting Association Football match outcomes. *Knowledge-Based Systems*, 36:322–339.
- Constantinou, A. C., Fenton, N. E., and Neil, M. (2013). Profiting from an inefficient Association Football gambling market: Prediction, Risk and Uncertainty using Bayesian networks. *Knowledge-Based Systems*, 50:60–86.
- Epstein, E. S. (1969). A scoring system for probability forecasts of ranked categories. *Journal of Applied Meteorology*, 8(6):985–987.
- FiveThirtyEight (2018). Club Soccer Predictions. <https://projects.fivethirtyeight.com/soccer-predictions/>. Last accessed on 2018-10-31.
- Football-Data (2018a). Historical Football Results and Betting Odds Data. <http://www.football-data.co.uk/data.php>. Last accessed on 2019-04-07.
- Football-Data (2018b). What is Football-Data? <http://www.football-data.co.uk/index.php#intro>. Last accessed on 2019-04-07.

- Gal, Y. (2016). *Uncertainty in deep learning*. PhD thesis, PhD thesis, University of Cambridge.
- Investopedia (2019). Weighted moving averages: The basics.
- Kelly, J. (1956). A new interpretation of information rate. *IRE Transactions on Information Theory*, 2(3):185–189.
- Langseth, H. (2013). Beating the bookie: A look at statistical models for prediction of football matches. In *SCAI*, pages 165–174.
- Markowitz, H. (1952). Portfolio selection*. *Journal of Finance*, 7(1):77–91.
- McCulloch, W. S. and Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *The bulletin of mathematical biophysics*, 5(4):115–133.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.
- Nielsen, E. S. and Sandøy, S. F. (2018). Improving Match Outcome Prediction Models In Association Football. <http://enielsen.tech/NielsenSandoy2018.pdf>.
- Pollard, R., Ensum, J., and Taylor, S. (2004). Estimating the probability of a shot resulting in a goal: The effects of distance, angle and space. *International Journal of Soccer and Science*, 2(1):50–55.
- Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- Rue, H. and Salvesen, O. (2000). Prediction and retrospective analysis of soccer matches in a league. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 49(3):399–418.
- Russell, S. J. and Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Malaysia; Pearson Education Limited,.
- Silver, N. (2009). A guide to ESPN’s SPI ratings. http://www.espn.com/world-cup/story/_/id/4447078/ce/us/guide-espn-spi-ratings. Last accessed on 2018-10-31.
- SMarkets (2019). Why you should consider the home advantage for football trading.

-
- Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.
- Vlastakis, N., Dotsis, G., and Markellos, R. N. (2009). How efficient is the european football betting market? evidence from arbitrage and trading strategies. *Journal of Forecasting*, 28(5):426–444.
- Xiong, J., Wang, Q., Yang, Z., Sun, P., Han, L., Zheng, Y., Fu, H., Zhang, T., Liu, J., and Liu, H. (2018). Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space. *arXiv preprint arXiv:1810.06394*.

