Haugård, Mii Erik Samyo
Vu, Kim Long

Predicting live matches

Beating the bookmaker using machine learning in live football

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

Master's thesis

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

NTNU
Norwegian University of Science and Technology

Haugård, Mii Erik Samyo
Vu, Kim Long

Predicting live matches

Beating the bookmaker using machine learning in live football

Master’s thesis in Computer Science
Supervisor: Helge Langseth
May 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

Football betting has increased in popularity over the past years. Several studies have attempted to predict the outcome of matches, both for fun and in an attempt to gain profit by using the predictions for betting. Most previous works have tried to predict the outcome of a match, prior to its start. This project will, on the other hand, focus on making predictions in an ongoing match.

In this project, neural networks are used in order to predict the next goal-scoring team in a live football match. Different set of features are used to detect which have the best predictive capabilities. These predictions are combined with several betting strategies, to decide if a bet should be placed or not, and how much to potentially bet, in an effort to generate a profit.

Through the project, we have achieved results that show it is possible to achieve high accuracy when predicting the next goal-scoring team in a football match. Further, the results show that using these predictions in order to beat the bookmakers can be achievable.

Sammendrag

Fotballbetting har i senere tid økt i popularitet. Det har vært utført mange studier som har prøvd å predikere utfallet av fotballkamper, både for moro skyld og for å prøve å profitere ved å bruke prediksjonene i bettingsammenheng. Mye av det tidligere arbeidet som er utført, har forsøkt å predikere utfall av en kamp, før kampstart. I motsetning til tidligere arbeid, skal dette prosjektet fokusere på å prøve å predikere i kamper mens de spilles.

I dette prosjektet skal vi bruke nevrale nett for å predikere neste lag som scorer i en kamp, mens den spilles. Forskjellig data er blir brukt sammen med det nevrale nettet, for å prøve å finne ut hvilke som har de beste prediktive egenskapene. Prediksjonene er videre brukt sammen med spillstrategier, for å bestemme hvor mye som eventuelt skal settes på et spill, i et forsøk på å få en positiv gevinst.

Resultatene fra prosjektet viser at det er mulig å predikere neste lag som scorer i en kamp, med høy nøyaktighet. Resultatene viser også at det er mulig å profitere ved bruk disse prediksjonene i bettingsammenheng.

Preface

This master's thesis is conducted by Mii Erik Samyo Haugrard and Kim Long Vu, at the Department of Computer Science (IDI) at Norwegian University of Science and Technology (NTNU) as a part of our masters' degree. This study was done in the period from January 2019 to May 2019, and is a continuation of our specialization project from the preceding semester. Throughout this period our supervisor, Helge Langseth, has assisted us with great insight in the line of work, guidance in the project and served as a good sparring partner.

We would like to thank Helge Langseth for being our supervisors in both our specialization project and master's thesis. We would also like to thank our families for their love and support.

Mii Erik Samyo Haugård & Kim Long Vu
Trondheim, May 29, 2019

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This initial chapter introduces the motivation for this work, the research questions, and the overall goal. Further, the structure of the thesis is presented.

1.1 Background and Motivation

Football is considered being one of the world's most popular sports, and a huge amount of money is involved in the game. This makes the betting market for football huge. In fact, football tops the list over the most betted on sports worldwide [van Lier, 2018].

This project is a continuation of our own specialization project [Haugård and Vu, 2018]. In that project, we looked at the state of the art in the field of predicting the outcome of football matches, and further, in some cases, applying these predictions in football betting. Most of these studies were about predicting the winner of a match, prior to its start.

The studies we researched in Haugård and Vu [2018] concerning live predictions did not apply their predictions for betting. Live predictions will be the focus in the project, where predictions are made while the match plays out, and further applying these predictions for betting. More specifically, the focus is on the betting market of predicting the next goal-scoring team. This means that at any given time in the match, it should be possible to make a prediction about which team is going to score the next goal, if any, and applying the prediction in betting.

1.2 Goal and Research Questions

The overall goal we try to achieve with this work is:

Goal *Use machine learning to predict the next goal-scoring team in a football match, in order to get an edge on bookmakers.*

To help us achieve this goal, the two following research questions are defined:

Research question 1 *What features are important when using machine learning to predict the next goal-scoring team?*

There are several potential features available that could be used in football predictions. We would like to find the ones that have the best predictive capabilities.

Research question 2 *How can the predictions be used to gain profit when betting?*

There are different money-management strategies when it comes to betting (these are described in Section 3.3). Even if the predictions are good, it is important to have a good betting system, in order to not go bankrupt after a couple of bets.

1.3 Thesis Structure

This project is divided into the following chapters:

Chapter 2: Background introduces the main theoretical aspects of this project. This chapter also looks at previous work done in the field of predicting ongoing football matches.

Chapter 3: Data and Model presents the model for predicting the next goal-scoring team, and the data used as input. In addition, the betting strategies used are presented.

Chapter 4: Experiments and Results describes how the experiments are set up, and presents the results from these experiments.

Chapter 5: Discussion looks further into the results from the previous chapter, analyses and discusses these.

Chapter 6: Conclusion and Future Work concludes the project, with regards to the research questions. Further, suggestions for future work are presented.

Chapter 2

Background

The following chapter presents a brief theoretic introduction to neural networks and money management in betting. Further, previous work in the field is presented.

2.1 Theory

This first section is a theoretic section that describes neural networks and money management. Some of the sections about the neural networks are also found in our specialization project [Haugård and Vu, 2018].

2.1.1 Neural Networks

Neural networks, also called artificial neural networks, are computer systems inspired by the use of neurons in the human brain. Each neuron produces an output when a linear combination of its inputs exceeds a threshold, which can be of a soft threshold or a hard threshold. A collection of several neurons that are connected together is called a neural network. The properties of the neurons and the topology makes the property of the network. The mathematical representation of the neuron can be seen in Figure 2.1. The neurons are called units or nodes in the networks and are connected via links between each other. A link from node i to node j serves as a way

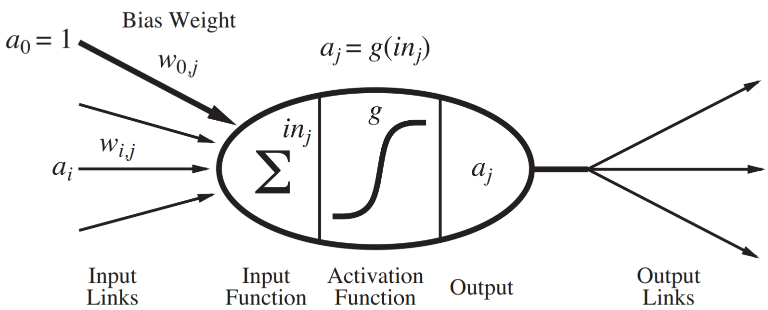


Figure 2.1: A mathematical representation of a neuron. The neurons output activation $a_j = g(\sum_{i=0}^{n} w_{i,j}\, a_i)$, where a_i is the output activation of unit i and $w_{i,j}$ is the weight on the link from unit i to this limit. [Russell and Norvig, 2016]

to propagate the activation a_i from i to j. Every link has a numeric weight $w_{i,j}$ to it, which determines the power and sign of the connection [Russell and Norvig, 2016].

To produce an output from the network, each unit j has to compute the weighted sum of its inputs:

$$in_j = \sum_{i=0}^{n} w_{i,j} a_i$$

And then use an activation function g to derive the output, where the activation function g is typically a hard threshold function or a soft threshold function, creating a perceptron or a sigmoid perceptron:

$$a_J = g(in_j) = g(\sum_{i=0}^{n} w_{i,j} a_i)$$

Layers is a central concept in neural networks, which forces each node to receive its inputs from other nodes in the preceding layer. A single-layer network will have its nodes connected directly from the input to the output, while a multilayer network will have one or more hidden layers that are connected to each other rather than directly to the output. Figure 2.2 illustrates a neural network with a single hidden layer.

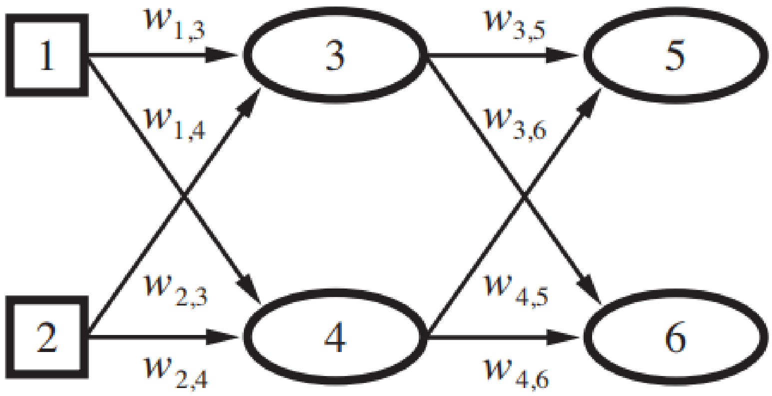


Figure 2.2: A neural network with two inputs, one hidden layer of two units, and one output unit. $W_{i,j}$ represents the associated weights from node i to j. [Russell and Norvig, 2016]

In order for the network to be able to learn from the predictions, a loss function is needed to asses how good the predictions are. There are different loss functions that calculate the model error, and cross-entropy is one of these. The cross-entropy is common to use with a classification model. Given two correct predictions, with a difference in their probabilities, the one with the higher probability will be considered more correct [Shibuya, 2018].

Optimizers are used to reduce the error. Adam is a popular optimizer, that calculates individual adaptive learning rates for each parameter. The Adam optimizer combines advantages from other optimizers, such as AdaGrad's ability to deal with sparse data and RMSProp's ability to deal with non-stationary problems. Adam is popular as it often achieves good results fast. [Brownlee, 2017].

After choosing the mathematical model for the neurons, a way of connecting them is required. Two different ways to connect the neurons are with a feedforward neural network, and with a recurrent neural network.

Feedforward neural networks

A feedforward network has only connections one direction, meaning that it creates an acyclic graph. Every node gets its input from an upstream node(s) and produces output to a downstream node(s). It has no internal state and no connections forming any cycles in the network. The feedforward network itself represents a function of its current input because of its lack of internal states [Russell and Norvig, 2016]. Figure 2.2 shows a simple feedforward network.

Recurrent neural networks

An RNN uses the output from a node and feeds it into its own input, thus creating a cycle. This creates the possibility for the activation levels of the network to create a dynamic system that might reach a stable state or chaotic state. Since these types of networks have nodes where the output may be dependent on previous inputs, we can say that RNN supports short-term memory [Russell and Norvig, 2016]. In other words, the network remembers the past and its decision are influenced by what it has learned from the past. On the other hand, the feedforward network also has the ability to remember, but only during training [Venkatachalam, 2019]. Figure 2.3 shows an illustration of the cyclic connection in an RNN.

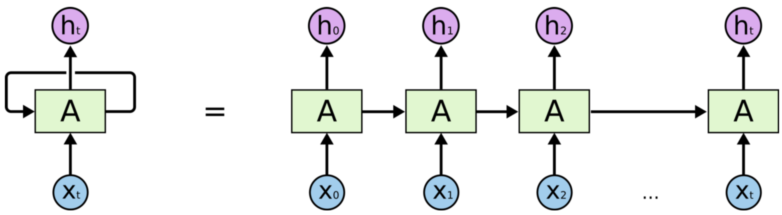


Figure 2.3: A simple RNN, showing the cyclic connection. [Donges, 2018]

The memory is stored in the hidden layers of the network and works as a context based on the previous inputs and outputs. The hidden layer is used together with weights to influence the produced output. This means that a produced output could be different based on previous input, thus using crucial information that is in the sequence of data, enabling the algorithm to find tendencies in the sequence itself [Venkatachalam, 2019; Donges, 2018].

Long short-term memory

Long short-term memory (LSTM) is an extension to the RNN and works as the memory in the hidden layer. An RNN can either have one or many (stacked) LSTM units in its network, where each unit remembers their inputs over a time period. LSTM units have similarities to regular computers as it can read, update and delete information stored in its memory. The memory can be considered as a gated cell, which is a cell that determines whether or not to store or delete data based on the level of importance on the information. The importance of information is assigned over time where important and unimportant information is learned. The learning happens through weights, which occurs while training the network [Donges, 2018].

There are three different gates in an LSTM unit: The input, the forget and the output gate. The input gate decides whether or not new input should enter the memory, the forget gate removes unimportant information based on its assigned importance, and the output gate decides if the stored information is affecting the output in the current time step [Donges, 2018]. A representation of an LSTM unit can be seen in Figure 2.4.

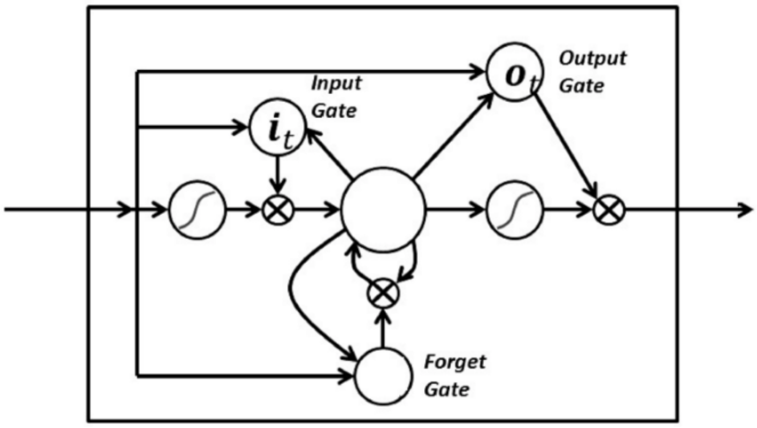


Figure 2.4: The internal model of an LSTM unit. [Donges, 2018]

Overfitting

As the network expands, introducing more parameters to the model, the problem with overfitting may occur. Overfitting can occur to all kinds of learners and causes the learner to produce an output that corresponds too similar to a particular set of data. The result of this is having a learner that fails to fit unseen data or produces future predictions inaccurately.

There are many techniques to prevent overfitting in neural networks. Early stopping is a form of regularization where the method stops the model from learning at given iterations when the model starts to overfit. A set limit could be specified, i.e. the maximum number of iterations during which no progress will be recorded. Exceeding this number will stop the model from learning. A validation set, which differs from the training set is often used to calculate the validation loss and validation accuracy. These values will then be saved and compared for each iteration until the validation loss is increasing and the validation accuracy decreasing [Skalski, 2018].

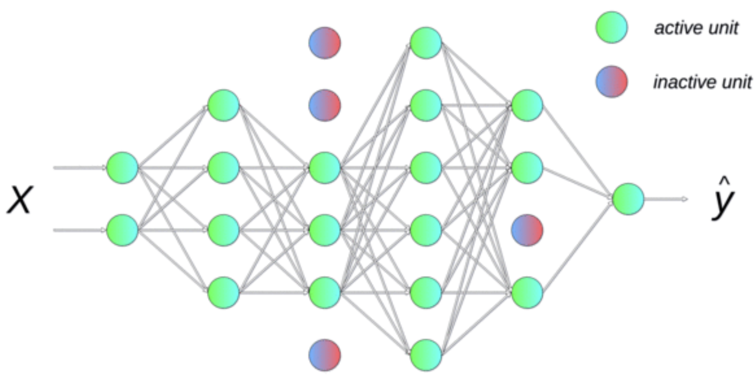


Figure 2.5: A neural network where four neurons have been ignored. The green dots represent active neurons, while the blue/red represents inactive neurons. [Skalski, 2018]

Dropout is another regularization method commonly used where every unit of the neural network, expect the units in the output layer, is given a probability of being ignored temporarily in the calculations. Then for each iteration, neurons get randomly selected to drop according to the probability that is defined. The result of this is that for each time we use the network, we get a smaller neural network, which tends to overfit less. Figure 2.5 illustrates a network where four neurons have been dropped.

2.1.2 Money Management

Expected value

To decide if a bet should be placed, it is important to calculate the expected value of the bet. A commonly used formula for this is [Langseth, 2013]

$$\textit{Expected value} = \textit{(Odds} \cdot \textit{probability of winning)} - 1.$$

In an imaginary match between Liverpool and Arsenal, where it is believed that Liverpool has a 40% chance of winning and a bookmaker offering an odds of 3, the expected value of a 1 unit would be

$$(3 \cdot 0.4) - 1 = 0.2.$$

Theoretically over time, one could expect to profit 20% on this bet. If the odds were to be adjusted however, this value could also be negative. Image the odds being 2 instead of 3 in the same game, with the same probabilities. The expected value would then be

$$(2 \cdot 0.4) - 1 = -0.2,$$

which indicates that this bet would not profit over time.

Calculating the expected value of a bet can give a good indication if a bet should be placed or not. However, a positive value does not guarantee a profit, nor does a negative guarantee loss. But consistently placing bets that have positive, and preferably high expected value gives a better chance of gaining profit.

Betting strategies

Two different bets can have the same expected value, but still have different probabilities and odds. One could argue that the bet with the highest probability is a better bet, but then one would have to have more money at the stake for the same potential reward. There are different strategies to this problem. Langseth [2013] presents different strategies that deal with the problem of how much one should place on a bet. The strategies output an amount c_i, that are to be placed on each bet i= 1, ..., n, where $c_i \leq C$ and C is the bankroll of the bettor. These outputs are based on the probability p_i and the odds o_i. Four of these strategies are:

- **Fixed bet:** Place the same amount of money on each bet, $c_i \propto 1$.
- **Fixed return:** Place the amount that makes sure that the same winnings can be won from each bet, $c_i \propto \frac{1}{o_i}$. This results in lower amounts placed on high-odds bets, and vice versa.
- **Kelly ratio:** Kelly [1956] proposed to use a decision-theoretic based approach as a strategy. In this setup, the utility of having a bankroll C after a bet is set to $\ln(C)$, meaning the utility of going broke is minus infinity. The expected utility of a bet is $p_i \cdot ln(C + (o_i - 1)c_i) + (1 - p_i) \cdot ln(C - c_i)$, which is maximized for $c_i \leftarrow C \cdot \frac{p_i o_i - 1}{o_i - 1}$. Langseth proposes a

modified version of this strategy, where the size of c_i cannot exceed a predefined value C_0, which is chosen to be considerably smaller than the bankroll C. Having a predefined limit ensures that the system does not lose too heavily during the first round and potentially get punished for this in later stages. Other versions of the Kelly strategy has also been used to limit the risk of losing the whole bankroll. One of these versions is called Fractional-Kelly betting [Australia Sports Betting, 2010]. Instead of having a cap C_0 as Langseth proposed, you multiply c_i by a number between 0 and 1. By using a lower number for this value, creates a more conservative strategy and minimizing the risk of going bankrupt in the long run. On the other, the potential winnings are also smaller than with regular Kelly.

- **Variance-adjusted:** The strategy proposed in Rue and Salvesen [2000] minimizes the difference between the expected profit and the variance of that profit. After placing a bet c_i, the difference is $p_i o_i c_i - p_i(1-p_i)(o_i c_i)^2$, which is minimized by choosing $c_i \leftarrow (2o_i(1-p_i))^{-1}$.

Yield

Yield is a metric that can be used when evaluating the profit made in betting. This is given by

$$Yield = \frac{\textit{Profit}}{\textit{Total units betted}} \cdot 100,$$

which gives a percentage representing the betting efficiency [Betacademy, 2019].

Return on investment (ROI) is also a commonly used metric for the same purpose. ROI is given by,

$$Yield = \frac{\textit{Profit}}{\textit{Initial bankroll}} \cdot 100,$$

where the initial bankroll is the number of units available. In contrary to when calculating yield, the ROI does not take into consideration the amount placed in bets.

2.2 Previous Work in the Field

This section presents previous work relevant to our field of work, which is predicting ongoing matches. We will look at Maher's contribution to this subject, and a couple attempts to predict the outcome of ongoing football matches. Only the most relevant studies are included in this section. For the interested reader, this was also reviewed in our specialization project, where we had a broader scope [Haugård and Vu, 2018].

2.2.1 Maher

A common approach to predict match outcome, without the use of machine learning, is to use Poisson distribution to measure the probability of the number of goals scored in a match. Maher is one of the first to investigate the use of the Poisson model with football scores. Earlier attempts have been made to fit a Poisson distribution to the number of goals scored in a match, but as Maher states in his paper, these earlier attempts rejected the use of the Poisson model in favor of another model [Moroney, 1951; Reep et al., 1971].

Maher works further with the Poisson model in his paper. To calculate the Poisson distribution for football results, the two opposing teams' attacking and defencing strength, α and β respectively, are used. Only the number of goals, the opposing teams, and the venue of the match are parameters for calculating the strengths values. When playing at home, team i's number of goals against team j is modelled as a Poisson variable, X_{ij}. The number of goals scored by team j is also modeled as a Poisson variable, Y_{ij}. Maher assumes these two, X_{ij} and Y_{ij}, to be independent of each other, meaning they can be evaluated as two separate games at each end of the pitch. Based on this, the outcome of a game between team i and team j is given by the distribution

$$P(X_{ij} = x, Y_{ij} = y|\alpha, \beta, k) = Poisson(x|k \cdot \alpha_i\beta_j) \cdot Poisson(y|\alpha_j\beta_i), \quad (2.1)$$

where k is the home field advantage assumed to be equal for all teams. An important note is that α and β do not vary over time.

The model was used on four English leagues over three seasons, giving a total of 12 data sets. Comparing the expected (calculated) with the observed (actual) goal frequencies, some systematic differences could be noticed. The model underestimated the probability for one and two goals to be scored and overestimated the probability that none and more than four goals would be scored. The difference between the expected and the observed frequencies are quite small, but added together would lead to a rejection of the model.

Further extensions of this model have been made to Maher's model. Dixon and Coles [1997] used results from 6629 league and cup matches from the top four English divisions from 1992 to 1995, and made a model that was able to generate score probabilities. They added a function $\tau(x, y)$ to Equation 2.1, that adjusted the probability for the low-scoring match results compared to Maher's model. Dixon and Coles also brings up the limitation with the model being static and that the attack and defence strength of a team is considered as constant through time by this model. This is not the case in reality, as a team's performance, considering both attack and defence, could vary from one time period to another. Dixon and Coles handles this by taking into account that recent matches reflect a team's current form better than matches earlier in history.

Even though Maher's paper was published in 1982, the model is still relevant today. With football being one of the most popular sports to bet on, many hobbyists have used similar kind of models to predict the outcome of matches [Ammon, 2016; Cronin, 2017]. These models calculate the goal probabilities for each team, which could be a good aid for different kinds of bets. Multiplying these probabilities and plotting the data into a table, as shown in Table 2.1, makes it easier to see what the probability for different outcomes is. E.g. is the probability that the home team does not score is below 4%.

Table 2.1: Results from prediction for a match using Poisson distribution. The blue field indicates the Poisson distribution for each of the teams while the green fields indicate home win, the red field indicates away win and the yellow field shows the probabilities for the different draw scores.

	Goals: Home team	0	1	2	3	4
Goals: Away Team	Probability for number of goals	3.65%	12.07%	19.99%	22.07%	18.27%
0	36.11%	1.32%	4.36%	7.22%	7.97%	6.60%
1	36.78%	1.34%	4.44%	7.35%	8.12%	6.72%
2	18.73%	0.68%	2.26%	3.74%	4.13%	3.42%
3	6.36%	0.23%	0.77%	1.27%	1.40%	1.16%
4	1.62%	0.06%	0.20%	0.32%	0.36%	0.30%

The two articles, by Ammon and Cronin, do not mention how their model's predictions did compare to the actual results. Even though they do not record any results, it seems like this model uses features that are necessary to predict a football match. The attacking and defencing strength for each team, which is the basis of this model, are calculated using other factors that could be useful in other models as well. These are the goal statistics during the season for each team, the number of goals scored and conceded in both home and away matches.

2.2.2 Live Predictions

The use of RNNs to predict the outcome of football matches has been explored by Pettersson and Nyquist [2017]. The data set consisted of matches from multiple seasons of many leagues from different countries (63 in total), in addition to tournaments that included teams from several countries (e.g Champions League). Information and events from these matches were used as input into the network. This includes lineups, the starting players' starting positions, goals, cards, substitutions, and penalties. Using an RNN ensures that the input data can have different sizes, which is important as the number of events in a match varies. Still, the input vector for each event has to have the same shape when fed into the network. This requires the information for different events to be merged into an input form that the network can handle. One of the methods Pettersson and Nyquist used to solve this problem is by using a one-hot vector containing all attributes for all events. Pettersson and Nyquist's model uses LSTM units, with a softmax classifier to represent a probability distribution over the three possible classes. The architecture of the model is shown in Figure 2.6.

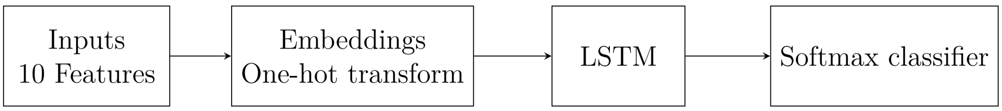


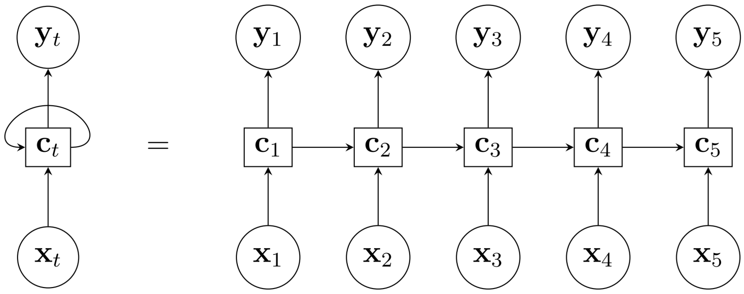
Figure 2.6: The RNN's high level architecture. [Pettersson and Nyquist, 2017]

Two different RNNs are compared, a many-to-one RNN and a many-to-many RNN. A many-to-many has an output at each timestep, while a many-to-one only has an output after the sequence of input (see Figure 2.7). Predictions were done using these RNNs, with 15-minute intervals for each match. The results from the prediction showed that the many-to-one performed better from the 60$^{\text{th}}$ minute and onwards, while a configuration of the many-to-many performed better up to this point. For each time step, the predictions are made by using events that have occurred up to this point. Table 2.2 shows the prediction accuracy at each time step for one of the many-to-many models and the many-to-one model. Initially, with only teams and lineups being known, the systems do not perform much better than random guessing. Both get higher

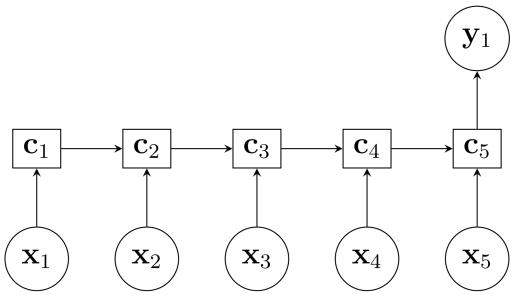
Table 2.2: Prediction comparison between the best configuration of the many-to-many model and the many-to-one model. Derived from Table 4.6 in Pettersson and Nyquist [2017].

	Prediction accuracy during match							
Model	0	15	30	45	60	76	90	Full Time
Many to many	**0.4396**	**0.4479**	**0.4705**	**0.5151**	0.5831	0.6797	0.8048	0.8868
Many to one	0.3335	0.3539	0.4151	0.5048	**0.6280**	**0.7409**	**0.8825**	**0.9863**

accuracy as the match nears the final whistle, as to be expected with more events fed into the system and less time left to score for the teams.



(a) Many-to-many RNN model, with an output at each time step.

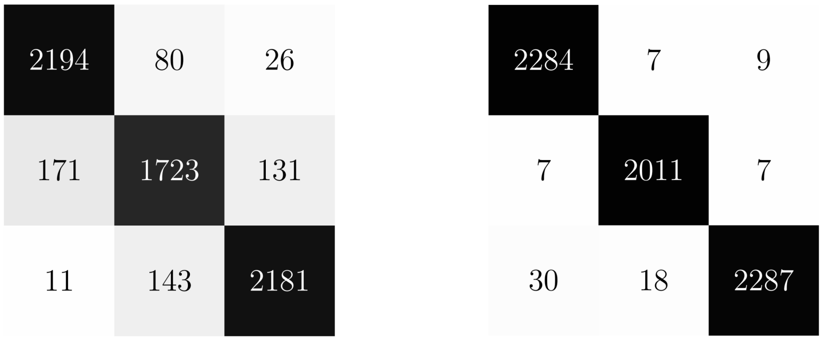


(b) Many-to-one RNN model, with an output only after the sequence of input.

Figure 2.7: The two different RNN models that were used in Pettersson and Nyquist [2017].

The test consisted of both classification and prediction, where the many-to-one model showed an overall higher accuracy than the many-to-many model. The many-to-many model was tested with different parameters, none performing better than the many-to-one model. The many-to-one approach calculates the accuracy at the end of the sequence, while the many-to-many averages over all the events. The classification results had a training accuracy of 100% and a test accuracy of 98% using the many-to-one model. The best test accuracy from a many-to-many model was 88%. However, the many-to-many model was "closer" to the correct answer when classifying wrong in the cases of home-win and away-win. Meaning in the cases of home win, only a few were classified as away win compared to draw. The same goes for the cases with away win; when classified wrong the system classified mostly as draw and not home win. With the many-to-one model, the model often classified the other team winning, rather than draw when classifying wrong. This is illustrated by the confusion matrices in Figure 2.8.

An approach related to live prediction is Boice's attempt to predict the outcome of the 2018 World Cup. In Boice [2018], a model is made based on creating Poisson distributions for each team and a matrix showing all the possible match scores with their probability. This makes it possible to find the pre-game probability of winning for each team. The difference with this model and the models based on Maher is that the attack and defence strengths used in this model are FiveThirtyEight's own variable called SPI rating. SPI rating is their own estimates of overall team strength, which are made up from match-based and roster-based ratings. These ratings are generated from data stored in their own database that contains matches that dates back to 1905. Based on the teams' ratings, a win/loss/draw probability matrix for a match is



(a) Confusion matrix from the many-to-many model.

(b) Confusion matrix from the many-to-one model.

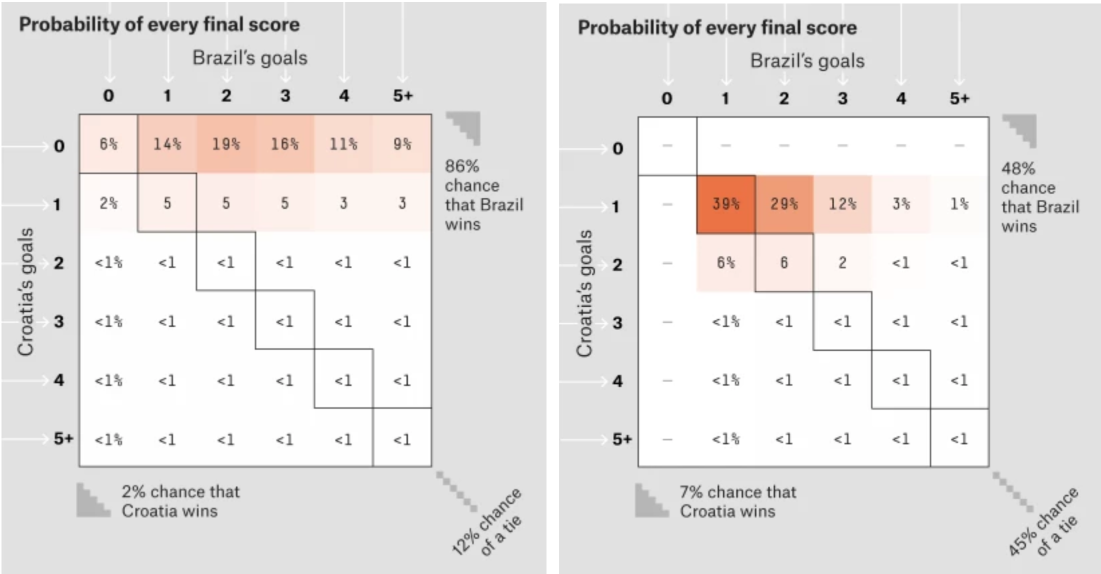
Figure 2.8: The two matrices show the classification results from the many-to-many model (a) and the many-to-one model (b). Each row corresponds to the actual results home win, draw, and away win respectively. The column represents, with the same order, what was predicted. A perfect classification would only have complete black cells along the diagonal, indicating 100% accuracy in the predictions. [Pettersson and Nyquist, 2017]

generated with the use of the Poisson distributions of expected goals for each team. Further, in order to forecast which team will win the World Cup, they used all the matrices with Monte Carlo simulations. This consists of simulating thousands of tournaments resulting in a winner based on how many times the team occurs in a winning simulation.

Live match predictions were also implemented in their work, which calculated each teams chances of winning, losing or drawing a match in real time. The live model works almost the same way as the pre-game predictions. First, the number of goals expected in the remaining time was calculated for each team. Then the Poisson distribution is created based on these numbers and fused together to create a matrix. In the end, the current score of the match is combined with the matrix, which gives the score probabilities in real time. Figure 2.9 shows how the probability matrix differences for the Brazil - Croatia match in the 2014 World Cup. Brazil was initially expected to be a clear winner, with a probability of 86%. After 65 minutes, with the score 1-1, this probability has now declined to 48%.

There are factors that need to be taken into consideration as the match plays out. Some important aspects Boice have considered, are that the scoring intensity at the end of a match is higher than at the beginning. Added time is also important, which extends the last period of the match. This is calculated by the number of bookings so far in the match, and whether or not it is a close match. Red cards also give a significant advantage and are taken into account. According to Boice, having one more player than the opposing team is worth three times more than a home-field advantage. In addition, after exploring the data, Boice discovered that a team that is down by a goal tends to have a higher scoring rate than what is indicated prior to the match.

Boice provided both pre-game and live predictions based on their model on their website throughout the World Cup, with the probabilities adjusted as the matches played out. As the article was written prior to the World Cup start, they have not concluded with how well their system performed. They do however have the predictions on which team will advance through each stage available on their website [FiveThirtyEight, 2018]. Comparing the predictions for the group stage and the actual results indicate that their system performed rather well. Of the 16



(a) Pre-game (b) After 65 minutes

Figure 2.9: Pre-game vs. live probabilities for the outcome in Brazil - Croatia, 2014 World Cup. How the probabilities of the final score changed after the game had played 65 minutes. [Boice, 2018]

teams that advanced from the group stage, Boice [2018] predicted 14 of them. Considering each team are in a group with three others, and in total two of them advances, 0.875% correct is at least better than random guessing (given the probabilities for different outcomes [Smith, 2017]). The two teams that were predicted to advance, but got knocked out of the group stage, are Germany and Poland. At least Germany was considered by many to have an easy process in advancing to the next stage. Of the 14 correctly predicted teams to advance, 11 of them were also predicted correctly whether they would finish first or second in their respective groups.

Chapter 3

Data and Model

This chapter presents the model for predicting the next goal-scoring team, and the data used as input. In addition, the betting strategies used are presented.

3.1 Data

The data set used for this project consists of matches from the last three seasons in the five biggest European leagues, which are the Premier League (England), La Liga (Spain), Serie A (Italy), Ligue 1 (France) and Bundesliga (Germany). Since one of the features used in the set, EGRT (Section 3.1.2), uses season averages to calculate its values, only matches in the second half of the season are included in the data set. The matches in the first half of the season are used to populate the tables which are used to calculate the season averages. Therefore, the 2017-2018 Premier League season matches before January 1st 2018 are used to populate the table, while the matches after this are included in the data set. The models (Section 3.2) use matches from the 2016-2017 and 2017-2018 seasons for training, while the matches from the current season are used for testing. That gives a total of 1880 matches in the training set, and 353 matches in the test set.

Finding good features with predictive power is important in prediction tasks. As Sportradar provided access to their API, most of the features are derived from the data gathered from this, which includes important match events (Section 3.1.1). Figure 3.1 shows Sportradar's API Map, which illustrates the information flow and how the data can be accessed. This was discussed in detail in Haugård and Vu [2018]. Data from Sportradar's API are also used to calculate the teams' EGRT values. The EGRT value is the number of goals the teams are expected to score in the remaining time of the match (Section 3.1.2).

3.1.1 Match Events

Pettersson and Nyquist [2017] used match events in their work, but these were limited to goals, cards, substitutions, and penalties. From Sportradar's *Match Timeline* API request, data on additional match events occurring during a match can be gathered as well. How detailed the data is presented depends on the leagues' coverage levels, but the matches included in the data set all include the following match events (if they occurred in the match): Score changes, goal kicks, throw-ins, shots saved, shots on target, shots off target, free kicks, injuries, corner kicks, offsides, cards, and penalties.

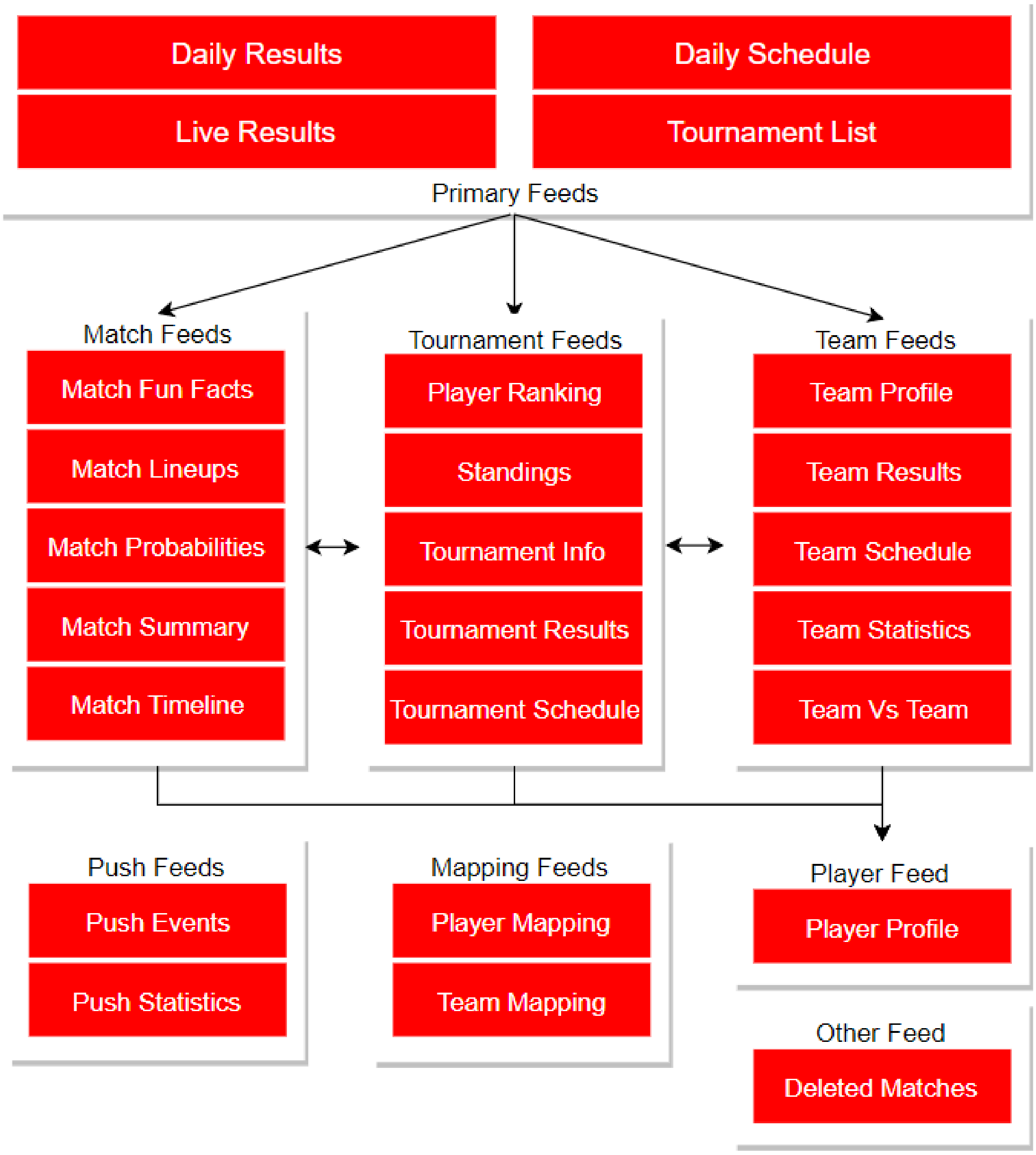


Figure 3.1: Sportradar's API Map. [Sportradar, 2019]

These events are tagged with either *home* or *away* depending on whichever team the event belongs to. The API request returns the events ordered by the sequence they occurred, and also with a time tag, called *match_time*, allowing us to know when in the match the event happened. Figure 3.2 shows two of the events from the first match of the English Premier League in the 2018-2019 season, a match between Manchester United and Leicester City.

```json
{
    "id": 447798530,
    "type": "throw_in",
    "time": "2018-08-10T19:38:14+00:00",
    "match_time": 39,
    "match_clock": "38:04",
    "team": "away",
    "x": 68,
    "y": 4,
    "period": 1,
    "period_type": "regular_period"
},
{
    "id": 447798864,
    "type": "shot_on_target",
    "time": "2018-08-10T19:39:22+00:00",
    "match_time": 40,
    "match_clock": "39:12",
    "player": {
        "id": "sr:player:111802",
        "name": "Pogba, Paul"
    },
    "team": "home",
    "x": 83,
    "y": 44,
    "period": 1,
    "period_type": "regular_period"
},
```


Figure 3.2: Sportradar API - Match events. [Sportradar, 2019]

3.1.2 EGRT

Based on the work from Section 2.2.1, we have calculated the teams' expected number of goals, which will be referred to as EG, based on the teams' attacking and defending strengths. The strengths are based on the teams' average goals and their belonging leagues' average. The attacking strength for the home team is given by

$$A_H = \frac{\textit{Home team's average goals scored home}}{\textit{The league's average home goals scored}},$$

where A_H is the home team's attacking strength. The defending strengths are calculated in a similar manner. The away team's defending strength is calculated like this:

$$D_A = \frac{\textit{Away team's average goals conceded away}}{\textit{The league's average away goals conceded}},$$

where D_A is the away team's defending strength. A team's attacking strength combined with the opposing team's defending strength are used when calculating the number of goals to be expected by a team in a match, and vice versa. The expected number of goals for the home team is therefore given by

$$EG_H = A_H \cdot D_A \cdot \textit{The league's average home goals scored.}$$

This EG value is how many goals one could expect a team to score in a match, based on the two facing teams' results in the current season. However, as we want to predict the next goal-scoring team at any given moment in a match, we would have to tweak this value to get the teams' expected number of goals in the remaining time (EGRT).

Pinnacle has one method for calculating this, by adjusting the initial expectation as time goes by [Pinnacle, 2019]. Their formula gives that 45% of the expected goals are scored in the first half, while the remaining 55% are scored in the second. This is naturally not the case for all teams, which is why we have chosen another approach.

This approach uses *Match Timeline* from Sportradar's API. As *Match Timeline* also logs the time for each event, making it possible to gather information about the match time of each goal scored by the different teams. Using this information, an EG value can be calculated for different time periods of a match. We chose to split the match into three equal periods, and generated tables for the different leagues with information on how many goals each team scored and conceded in each of the periods. The table for the English Premier League from the 2018-2019 season, until March, can be seen in Table 3.1.

The EG values and the EGRT values calculated prior to a match will be the same. The EGRT values will, however, decrease as the match plays out. In a hypothetical match between Tottenham and Arsenal, where Tottenham plays at home, the teams' EGRT values for the remaining 45 minutes will be as following:

Tottenham's EGRT

Considering we want to find the EGRTs after 45 minutes being played, we will only use the number of goals in the last two time periods to calculate Tottenham's attacking strength playing at home, and Arsenal's defending strength playing away.

$$A_{31-60} = \frac{6/13}{131/279} = 0.9830$$

$$D_{31-60} = \frac{10/13}{131/279} = 1.6383$$

The values above represent the home team's attacking strength and the away team's defending strength, in the time period 31-60 minutes. These values are generated from the goals scored and conceded in the second 30-minutes period we have split the matches in. We assume a goal can happen whenever within these periods with the same probability. The EG value is therefore multiplied with 15/30, as we only want to find the value for 15 minutes of the 30-minutes period.

$$EG_{46-60} = A_{31-60} \cdot D_{31-60} \cdot \frac{131}{279} \cdot \frac{15}{30} = 0.3781$$

Further, the values for the last period (61-90 minutes) is:

$$A_{61-90} = \frac{11/13}{184/279} = 1.2830$$

$$D_{61-90} = \frac{9/13}{184/279} = 1.0497$$

$$EG_{61-90} = A_{61-90} \cdot D_{61-90} \cdot \frac{184}{279} = 0.8882$$

This gives that following number of expected goals for the home team after 45 minutes has been played:

$$EGRT_{45} = EG_{46-60} + EG_{61-90} = 1.2663$$

Table 3.1: Premier League table from the 2018-2019 season showing how many goals each team have scored and conceded in each of the three periods when splitting a match into three equal periods.

Team		Matches	Scored	Conceded
AFC Bournemouth	Home:	14	8 - 8 - 10	4 - 6 - 8
	Away:	14	3 - 7 - 3	10 - 13 -12
Arsenal FC	Home:	15	9 - 9 - 17	5 - 2 - 5
	Away:	13	9 - 9 - 7	7 - 10 - 9
Brighton & Hove Albion FC	Home:	13	5 - 6 - 5	3 - 5 -7
	Away:	14	4 - 4 - 5	8 - 8 - 10
Burnley FC	Home:	13	5 - 8 - 4	9 - 5 - 9
	Away:	15	2 - 5 - 7	6 - 8 - 13
Cardiff City	Home:	15	4 - 4 - 8	10 - 9 - 12
	Away:	13	1 - 0 - 8	5 - 6 - 13
Chelsea	Home:	14	7 - 7 - 14	1 - 6 - 2
	Away:	13	4 - 8 - 7	7 - 6 - 7
Crystal Palace	Home:	14	2 - 3 - 5	1 - 8 - 6
	Away:	14	1 - 10 - 11	3 - 7 - 13
Everton FC	Home:	14	5 - 8 - 8	4 - 10 - 7
	Away:	14	6 - 2 - 10	2 - 9 - 7
Fulham	Home:	13	2 - 8 - 7	11 - 5 - 10
	Away:	15	3 - 3 - 3	14 - 9 - 14
Huddersfield Town	Home:	15	3 - 2 - 2	6 - 11 - 5
	Away:	13	2 - 3 - 3	9 - 7 - 12
Leicester City	Home:	14	6 - 3 - 7	7 - 5 - 6
	Away:	14	1 - 8 - 9	6 - 7 - 8
Liverpool FC	Home:	14	13 - 12 - 15	1 - 3 - 3
	Away:	14	5 - 6 - 13	2 - 2 - 4
Manchester City	Home:	15	19 - 16 - 15	3 - 6 - 2
	Away:	13	8 - 11 - 6	1 - 3 - 5
Manchester United	Home:	13	9 - 5 - 11	3 - 5 - 8
	Away:	15	6 - 15 - 9	9 - 4 - 7
Newcastle United	Home:	15	5 - 4 - 7	7 - 3 - 8
	Away:	13	7 - 3 - 0	3 - 5 - 8
Southampton FC	Home:	14	5 - 7 - 5	3 - 10 - 8
	Away:	14	4 - 6 - 3	9 - 9 - 7
Tottenham Hotspur	Home:	13	7 - 6 - 11	1 - 5 - 6
	Away:	15	10 - 13 - 8	4 - 4 - 9
Watford	Home:	13	3 - 5 - 9	2 - 10 - 6
	Away:	15	8 - 4 - 10	7 - 4 - 11
West Ham United	Home:	14	4 - 7 - 10	6 - 6 - 8
	Away:	14	7 - 3 - 4	5 - 7 - 9
Wolverhampton Wanderers	Home:	14	3 - 3 - 14	8 - 3 - 8
	Away:	14	4 - 3 - 8	7 - 3 - 6

Arsenal's EGRT

Similarly, Arsenal's EGRT at minute 45 is calculated the following way:

$$A_{31-60} = \frac{9/13}{123/279} = 1.5704$$

$$D_{31-60} = \frac{5/13}{123/279} = 0.8724$$

$$EG_{46-60} = A_{31-60} \cdot D_{31-60} \cdot \frac{123}{279} \cdot \frac{15}{30} = 0.3020$$

$$A_{61-90} = \frac{7/13}{134/279} = 1.1211$$

$$D_{61-90} = \frac{6/13}{134/279} = 0.9610$$

$$EG_{61-90} = A_{61-90} \cdot D_{61-90} \cdot \frac{134}{279} = 0.5174$$

$$EGRT_{45} = EG_{46-60} + EG_{61-90} = 0.8194$$

The EGRT values can be calculated at any given time during a match between two teams in the same league, and further, be used as features. The example showed a calculation of the EGRT values for a match between Tottenham and Arsenal, based on the number from the leagues up until the 1st of March. The actual match between the two teams that played out on the 2nd of March ended up with only Tottenham scoring in the last 45 minutes. [Premier League, 2019]

3.1.3 xG

In addition to the data from Sportradar, the expected goals metric (xG) is gathered from Understat and used as a feature as well. This was also reviewed in detail in our specialization report [Haugård and Vu, 2018]. This metric gives an indication of how great the chance is for a given goal-scoring attempt to be converted into a goal. The xG metric has the possibility to reveal that a team should have scored more goals statistically, either during a single match or over an entire season. We have gathered the xG data for the matches from Understat, as this was something not available from Sportradar. Understat keeps detailed xG data on the matches that have been played in the top European leagues. The xG value of a shot is determined by a neural network trained on over 100.000 shots with over 10 parameters for each shot [Understat, 2019b].

According to Understat, the xG value usually gives a better picture of two teams' performance and how a match actually played out rather than the final match score, especially in a low-scoring game. Imagine a match where the half-time score is 1-0, but the xG values for the same match is actually 0.65 - 3.43. This indicates that the leading team currently has had a very good pay off on the few chances they had, while the losing team has had a higher amount of chances without any of them resulting in a goal. If the teams keep up the same frequency of chances in the second half, one could expect that the losing team would get an equalizer, at the very least.

Figure 3.3 shows a chart with the xG values for each minute of the match between Eibar and Celta Vigo, played on the 3rd of March. The figure shows that Eibar clearly created more chances than Celta Vigo. At the 80th minute, Eibar's xG value is 1.86, indicating that they statistically should have at least scored one goal. At this point, the score was still 0-0. However, Eibar finally scored the match's only goal six minutes later, where the new score now better reflects the teams' effort throughout the match.

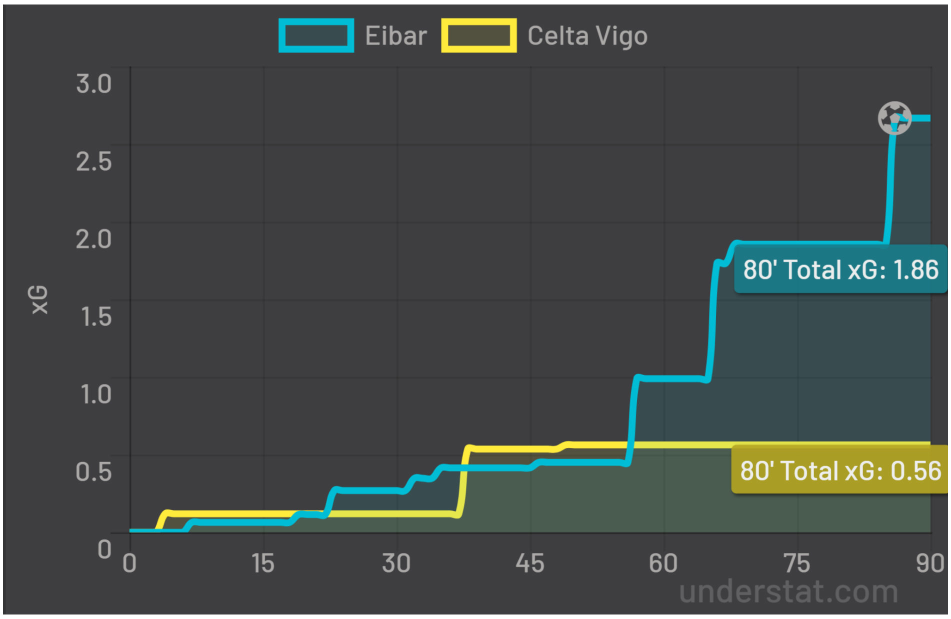


Figure 3.3: Understat - xG values' development during a match between Eibar and Celta Vigo. [Understat, 2019a]

3.2 Neural Networks

To solve the task of predicting the next goal-scoring team, two different neural networks have been created. One feedforward neural network and one RNN. Different network structures were later tested in order to find the optimal set of hyperparameters, which would give the best predictions.

3.2.1 Feedforward Neural Network

Feedforward networks require the input to be the same size, meaning the input vectors fed into the network need to be of the same length. This is done by using a vector that includes all the features. This often leads to the input being very sparse, as all the events do not occur every match, but all the features could be included in the same vector. Figure 3.4 shows an example of an input vector from a match at the 75th minute where EGRT, score changes, shots on target and match time are used as features. The match events each has four elements in the vector: Home team's match total, away team's match total, home team's number of the event's occurrences in the last 15 minutes and away team's number of the event's occurrences in the last 15 minutes. So even if some matches have different types of events, the size of the input vectors will remain the same, which is required in a feedforward network.

The cross-entropy is used as the loss function, while the Adam optimizer is used to reduce the error. As mentioned in Section 2.1.1, these are common to use with classification tasks. Further, Softmax is used as the activation function in the output layer, with three possible

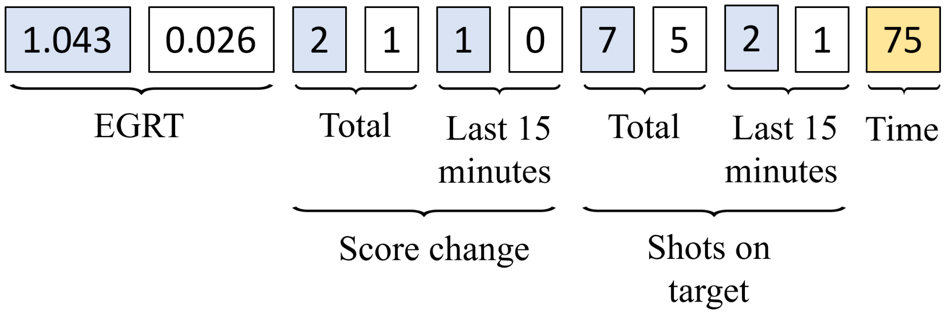


Figure 3.4: Input vector for the feedforward network, with some of the features used. The blue-colored background indicates the home team's values.

classes. This gives a vector with the probability for each of the three outcomes. When training, the probabilities from the predictions are compared to the target vector y, which is one of following depending on which team scores the next goal:

$$y = \begin{cases} [1,0,0] & \text{Home team scoring next goal} \\ [0,1,0] & \text{No more goals in the match} \\ [0,0,1] & \text{Away team scoring next goal} \end{cases}$$

The accuracy is calculated by comparing the target vector with the output vector. Given a prediction vector [0.3, 0.2, 0.5] in a case where the away team is the next goal-scoring team, the prediction would be considered correct.

To avoid overfitting, early stopping is used. The models stop training when the validation loss stops decreasing, with a threshold of five iterations. The validation happens after every epoch, where the model is tested on the validation set, which is a part of the training data set that has not been trained on.

3.2.2 Recurrent Neural Network

An RNN was also implemented in an attempt to learn the match dynamics from the sequence of match events better. Previous work with the use of LSTM cells in RNN has shown promising results, as we have seen in Section 2.2.2. The RNN developed in our case uses LSTM cells, with a different number of LSTM cells. The data set used for the RNN consist of the same matches that were used with the feedforward network. However, the data had to be preprocessed differently before feeding it into the network. Instead of summing up the number of times an event has happened, the events are represented by a one-hot vector and fed into the network as a sequence, in the order the events occur. As there are a different number of events in matches, the number of input vectors for a match can vary. However, the length of each vector is constant. Figure 3.5 shows how the match events are represented in the input vector, where the first event is a throw in for the home team and the last is a free kick for the away team. Match time, EGRT and xG values are also used as features in the RNN and represented the same way as in the feedforward network.

As with the feedforward network, the cross-entropy loss function is used together with the Adam optimizer. The output layer has a Softmax activation function giving a prediction vector of [*Probability home team scoring next goal*, *Probability no more goals*, *Probability away team scoring next goal*]. The prediction is compared with the target vector before the loss is calculated and

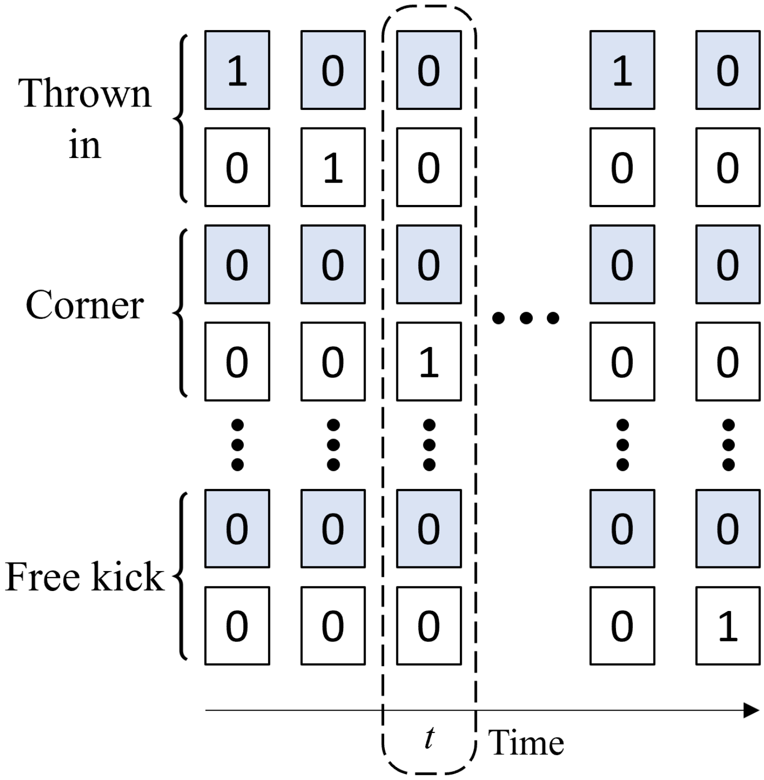


Figure 3.5: The sequence of match events represented by one-hot vectors (vertical) the way they are fed into the RNN. At time t we can see that the away team got a corner. The blue-colored background indicates the home team's values.

back propagated to modify the weights. The highest value of the three predicted probabilities is considered the produced prediction from the model.

Similar to the feedforward network, early stopping is used to avoid overfitting. In addition to early stopping, dropout is also used to help avoid this problem. This means that each node in the last LSTM layer has a chance of being left out during the training, thus reducing the chance to overfit.

3.3 Betting Simulator

The four different betting strategies from Section 2.1.2 are implemented in order to decide how much to place on each bet in betting simulations. This is based on the probability from the predictions produced by the neural networks and the odds provided by Sportradar. For the Kelly strategy, the Fractional-Kelly strategy is used. Each strategy is implemented to run its own betting simulation, which consists of matches from the current season. Bets are placed on the most likely outcome according to the network, and only if the bets are feasible, meaning the expected value is positive. Each simulation starts with an initial bankroll, which gets updated for each bet.

The four strategies only consider how much to place on single bets, and does not consider multiple bets in the same match. To test this we have created our own strategies to test the possibility to place several bets on the same outcome. The strategies are based on the fluctuations of odds and calculated probabilities, but also considering the amount that is already placed in the same bet. In other words, if a bet already has been placed early in a match, should one place more bets later in the game if one of the team seems to be more likely to score than initially predicted? Based on the Kelly and the variance-adjusted strategies, we propose two strategies for this purpose:

- **Continuous Kelly:** With the original version, the expected utility of a bet is $p_i \cdot ln(C + (o_i - 1)c_i) + (1 - p_i) \cdot ln(C - c_i)$. As we now also want to consider how much to place on a bet, while also taking the amount X already placed on a bet into account, the modified expression for the expected utility is $p_i \cdot ln(C + (o_i - 1)(c_i + X)) + (1 - p_i) \cdot ln(C - (c_i + X))$. This is maximized for $c_i \leftarrow C \cdot \dfrac{p_i o_i - 1}{o_i - 1} - X$.
- **Continuous Variance-adjusted:** This strategy is also modified in a way that includes the amount X that has already been placed. We still want to minimize the difference between the expected profit and the variance of that profit, which now is $p_i o_i(c_i + X) - p_i(1 - p_i)(o_i(c_i + X))^2$ after placing a bet c_i. This is done by choosing $c_i \leftarrow (2o_i(1 - p_i))^{-1} - X$.

Both of these are similar to their original expression, with the difference of X now being subtracted. For these two strategies, two additional factors need to be taken into consideration:

1. **Betting frequency -** How often during a match one should consider a new bet.
2. **Change in predicted outcome -** What to do if the neural network predicts an outcome to be the most likeliest, which differs from the one that has been betted on.

For changes in the predicted outcome, we have implemented two different approaches. The first approach considers bets on other outcomes as lost bets, and bets on the new recommended outcome without considering how much has been placed on other outcomes. This means that the potential winnings from each match will be smaller, as some bets will be against each other. The loss will also be smaller with this approach. The second approach goes all in on the new prediction. In addition to betting the amount decided by the strategies, we will also place an additional bet that cancels out previous bets on the other outcomes. For instance, in a match where 5 units have been placed on the home team scoring the next goal, but the network now predicts that the away team is the most likely next goal-scoring team with a probability of 0.55. With an odds being 2, the total amount to bet on the new outcome, using the variance-adjusted strategy is

$$(2 * 2(1 - 0.55))^{-1} + \frac{5}{2} = 3.06,$$

where the second term is to cancel out previous bets.

Both approaches are implemented with both the Kelly and the variance-adjusted strategies, giving a total of 4 different strategies for betting multiple times in a match:

- **Continuous Kelly 1:** Considers bets on other outcomes as lost bets using the Kelly strategy.
- **Continuous variance-adjusted 1:** Considers bets on other outcomes as lost bets using the variance-adjusted strategy.
- **Continuous Kelly 2:** Goes all in on the new prediction using the Kelly strategy.

- **Continuous variance-adjusted 2:** Goes all in on the new prediction using the variance-adjusted strategy.

Chapter 4

Experiments and Results

This chapter starts by describing the experimental plan and the experimental setup, which describes how to test the model and validate the predictions made by the model. Further, this chapter presents the results from the experiments.

4.1 Experimental Plan

As stated in Section 1.2, the main goal with this project is:

Goal *Use machine learning to predict next goal-scoring team in a football match, in order to get an edge on bookmakers*,

and to achieve this, we first need a model that has a satisfyingly high accuracy when making predictions. Further, the model will be used in betting simulations to verify if it is possible to profit from the predictions.

Our first research question is

Research question 1 *What features are important when using machine learning to predict the next goal-scoring team?*

To answer this, we will train different models using different feature sets. As we want to predict the next goal-scoring team at any given point in the match, we will also test how well an RNN performs compared to a feedforward network. Considering the RNN being able to take the sequence of match events as input, we might learn that certain sequences often lead to goals.

To evaluate the models, we will compare the validation accuracies of the models. These accuracies are also to be tested against several baselines, and should obtain values that are better than these. These baselines are the accuracy when always choosing the home team to score the next goal, the accuracy when always choosing that there will be no more goals, the accuracy when always choosing the team with the highest EGRT, and the accuracy when always choosing the team with the highest xG-value.

Further, our second research question is

Research question 2 *How can the predictions be used to gain profit when betting?*

Therefore, the models that achieved the highest validation accuracy in each of the cases are to be used in betting simulations on the test set. The predictions made by the models are tested against actual odds from Sportradar, in order to validate the reliability of the predictions. Each of the cases will get evaluated with different betting strategies described in Section 3.3. These strategies are also tested against the yield achieved when always betting on the home team, always betting that there will be no more goals, always betting on the team with the highest EGRT, always betting on the team with highest xG value, and always betting on the team with the lowest odds.

4.2 Experimental Setup

Six different cases are to be tested with both networks, where different combinations of features are used in the different cases. Three of the cases include matches only from Premier League, while the remaining cases include matches from all five leagues. If there are any differences in how the different features interrelate with who the next goal scorer is in the different leagues, then having a model training on only one league could give a higher accuracy. The specification for each of the cases can be seen in Table 4.1. As discussed in Boice [2018], time is an important factor when predicting ongoing matches. For instance, the scoring intensity at the end of a match is higher than at the beginning. Thus, in addition to the features listed in the table, all cases include the match time as a feature.

Table 4.1: Overview of the features included in the different cases.

	Premier League	**Other 4 leagues**	**EGRT & xG**	**Events**
Case 1	Yes	Yes	Yes	Yes
Case 2	Yes	Yes	Yes	No
Case 3	Yes	Yes	No	Yes
Case 4	Yes	No	Yes	Yes
Case 5	Yes	No	Yes	No
Case 6	Yes	No	No	Yes

The experimental setup consist of two main parts for each case, testing different combinations of network parameters that give good predictions, and the betting simulation part. As mentioned in Section 3.1, data from the five big European Leagues will be used, which are the Premier League (England), La Liga (Spain), Serie A (Italy), Ligue 1 (France) and Bundesliga (Germany). The models are trained on matches from the second part (starting January 1st) of the season, in the 2016-2017 and 2017-2018 seasons. The training set consist of 1880 matches in total. The test set consist of the matches played in January and February in the 2018-2019 season, a total of 353 matches. Testing different combinations of network parameters are done on the training set, while the betting simulation part is executed on the test set.

4.2.1 Finding the Best Network Parameters

For each case, the goal is to find the best network parameters for both the feedforward network and the RNN. The size of the input layer varies with the number of features used as input, while the output layer always consist of three nodes as there are always three possible outcomes. However, the other parameters in the networks can be adjusted. For the feedforward network, these are: Different number of hidden layers and number of nodes in these layers, hidden layer's activation function, learning rate and batch size. For the RNN, the parameters are: Number of LSTM layers, number of nodes in the hidden layers and learning rate.

For the feedforward network, the training set consists of data from four randomly chosen time points in each match, which gives a total of 1880 times 4 cases in the training set. The data set for the RNN consist of the same matches, but each match event is represented by its own input vector (see Figure 3.5). These are fed into the RNN sequentially for each match. To find the best combination of feature set and network parameters for both networks, 20% of the training set are used as validation set. An important notice is that this is 20% of the matches, and not 20% of the cases. This means that for the feedforward network, if one of the four cases from a match is in the validation set, the remaining three cases from the same match will also be in the validation set. The validation accuracy is used as the criteria when comparing the different setups. Another option would be to use actual odds and run betting simulations on the validation set, and select the setup that achieved the highest yield with the validation set. However, we do not have odds for the matches in the validation set available, which leaves using the validation accuracy as the only option. Hence, the setup achieving the highest validation accuracy for each case are used in the betting simulations.

4.2.2 Betting Simulation

With regard to the second research question, we will run betting simulations based on the predictions made by the networks. The seemingly best combinations network parameters are going to be used for the simulation for each of the cases. As mentioned previously, the test set used for the betting simulation consists of the matches from January and February in the current season. This consist of 353 matches when all five leagues are included, and 79 when only Premier League matches are included. The odds for these matches are provided by Sportradar.

Three different simulation experiments based on the predictions will be done. Each simulation starts with an initial bankroll of 100 units where the first two simulations are using the strategies for single bets stated in Section 3.3, with 0.25 as the multiplier in the Fractional-Kelly strategy.

- **Betting simulation 1:** The first simulation places a bet at one random time point in each of the matches.
- **Betting simulation 2:** The second one is too see whether there are any time points in the match where it is easier to gain a profit from than others. Thus, running six times at six fixed time points, which are the 15th, 30th, 35th, 60th, 75th and 90th minute.
- **Betting simulation 3:** The last simulation experiment deals with the possibility to place multiple bets during a single match. For instance, this occurs when a bet already is placed at the 20th minute, but at minute 25 the prediction has changed, indicating that you should have betted more, or less, than you actually did five minutes ago. The four different strategies proposed in Section 3.3 for placing multiple bets in a single match are used for this. The betting frequency is chosen to be five minutes, meaning a new bet is considered with five minutes apart.

Further, each simulation is executed five times, each time with a different network instance, trained with the best combination of parameters. The 20% of the training set that are used as the validation set is random for each of the five instances. This is to get a better indication of the robustness of the network. The worst, the best and the average results are presented.

4.3 Experimental Results

This section presents the results of the experiments for each of the six cases. For each case, a table showing the accuracy with different network parameters and charts displaying the results from the betting simulations are presented.

4.3.1 Baselines

To evaluate the models and their predictions, we will compare them to several baselines. The baselines represent the lowest values that our models should obtain. As mentioned in Section 4.1, these baselines are based on:

- **Home team:** Always choosing the home team to be the next goal-scoring team.
- **No goals:** Always choosing that there will be no more goals in the match.
- **EGRT:** Always choosing the team with the highest EGRT value to be the next goal-scoring team. If the teams have the same EGRT, the home team will be chosen.
- **xG:** Always choosing the team with the highest xG value to be the next goal-scoring team. If the teams have the same xG, the home team will be chosen.
- **Odds:** Always choosing the team with the lowest odds to be the next goal-scoring team. If the teams have the same odds, the home team will be chosen. This is only used for the yield baseline, as we do not have the odds for the training set available.

The accuracies and yields for the baselines are shown in Table 4.2. These are generated from the entire training set, where four random time points are chosen in each match. The yields are generated from the test set, where random time points are chosen in each match. The fixed bet strategy from Section 3.3 is used, as it does not require probabilities. The accuracies are between 0.41 and 0.44, where always choosing the team with the highest EGRT value gave the highest accuracy. The baselines for the yields had a greater variety than the accuracy. Placing bets based on the EGRT and xG values did not generate a positive yield, while always betting on the home team gave the highest yield at 6.23.

Table 4.2: The accuracies and yields for the baselines.

	Home	**None**	**EGRT**	**xG**	**Odds**
Accuracy	0.41	0.28	0.44	0.42	-
Yield	6.23	5.43	-4.50	-8.24	4.62

4.3.2 Case 1

Case 1 consists of data from all the leagues and uses all the available features as input.

Network parameters

For the first part, we wanted to find the combination of network parameters that gave the highest validation accuracy. The accuracy should be at least as high as the ones achieved by the baselines, where the highest was 0.44. Table 4.3 shows the ten combinations of network parameters that obtained the highest accuracies, for both the feedforward network and the RNN. RNNs are indicated with a blue-colored background and no batch size specified, as they are not trained in batches. An RNN with 4 LSTM layers, 32 units in each layer, and a 0.001 learning rate gave the highest validation accuracy. This network had a validation accuracy of 0.53499, which is significantly higher than the baselines.

Table 4.3: Best network parameters with Case 1. The blue-colored background indicates an RNN.

Number of hidden layer(s) and nodes in each layer	Hidden layer(s) activation	Learning rate	Batch size	Validation accuracy
4 layers 32 nodes	LSTM with tanh	0.001	-	**0.53499**
2 layers 32 nodes	LSTM with tanh	0.001	-	**0.53386**
6 layers 64 nodes	LSTM with tanh	0.001	-	**0.53202**
1 layer 57 nodes	relu	0.002	256	**0.53158**
1 layer 57 nodes	relu	0.001	32	**0.53001**
1 layer 30 nodes	relu	0.002	32	**0.52979**
4 layers 64 nodes	LSTM with tanh	0.001	-	**0.52861**
4 layers 256 nodes	LSTM with tanh	0.001	-	**0.52843**
2 layers 64 nodes	LSTM with tanh	0.001	-	**0.52828**
1 layer 57 nodes	relu	0.001	256	**0.52708**

Betting simulation 1

Figures 4.1 to 4.4 show how the bankroll develops during the betting simulations, using the different betting strategies while betting at random time points. The yields from the simulation are showed in Table 4.4. None of the average yields are better than the baseline yields achieved by always betting on the home team. For Case 1, the best average and the only positive result was with using the variance-adjusted strategy.

Table 4.4: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-6.76	-5.70	-11.72	-3.03
Maximum	7.95	7.54	2.62	7.66
Average	-0.15	-0.24	-4.43	1.11

The figures show three graphs each. The red and green graphs represent the bankroll development of the runs that gave the worst and best yields, respectively. These are the worst and best runs, according to their final bankroll value, out of the five runs. The multiple runs were performed in order to test the robustness of the network. A large gap between these charts indicates a less robust network, and the performance depends largely on how the training data

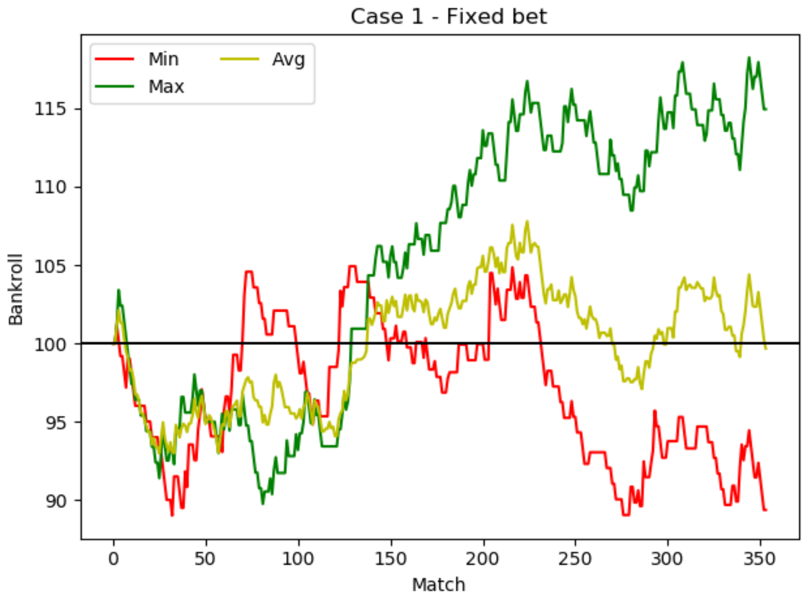


Figure 4.1: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

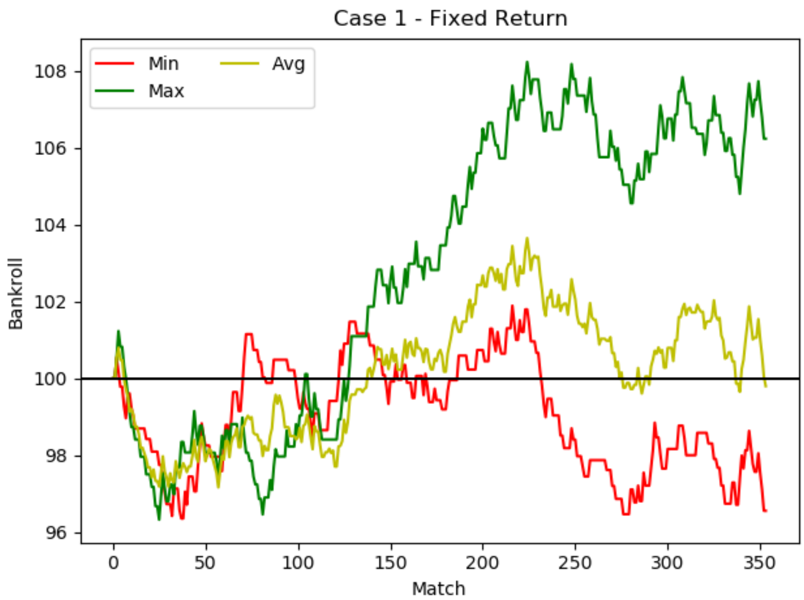


Figure 4.2: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

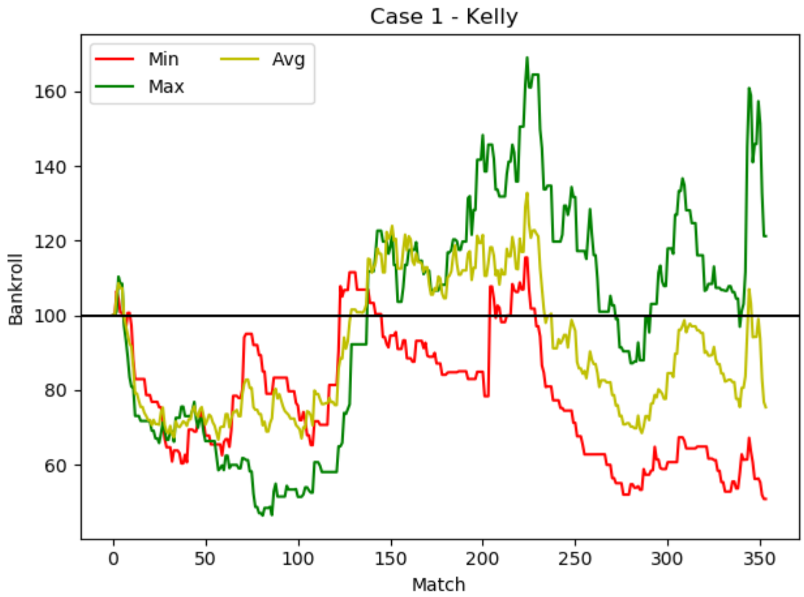


Figure 4.3: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

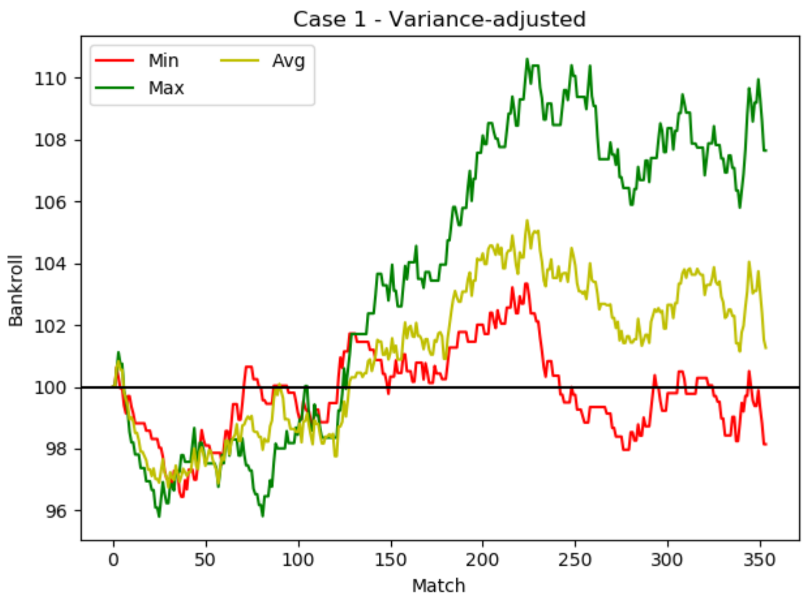


Figure 4.4: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

was split, which can cause small differences in the predicted probabilities. These small differences can be enough to decide if a bet is considered feasible or not. We can see from the figures that the difference in the final bankroll values are big with all the betting strategies. This could be acceptable if the worst run also generated a positive yield, but this is not the case.

The yellow graphs indicate the five simulations' average bankroll value after each match. To consider using a betting strategy, the average run should generate a positive yield. This is only the case with the variance-adjusted strategy.

Betting simulation 2

Table 4.5 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. This was explored in order to see if there were any specific time points during the matches that effortlessly generated positive yield. With Case 1, placing bets at the end of the match, specifically at the 75th and 90th minute, gave positive yields independent of the betting strategy used. Placing bets prior to this gave, without exception, negative yields. Only placing bets at the 90th minute also achieved yields that are higher than the baselines. The highest yield was achieved using the Kelly strategy, at the 90th minute in the matches.

Table 4.5: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	-0.57	-1.42	-1.18	-1.43
30	-2.39	-2.68	-7.04	-2.45
45	-4.29	-3.87	-14.01	-3.69
60	-7.15	-3.65	-22.36	-2.90
75	6.26	6.19	2.14	6.22
90	48.25	48.69	54.11	48.63

Betting simulation 3

Table 4.6 shows the results from the simulations where multiple bets could be placed in each match. The best average yield was achieved using the variance-adjusted strategy where previous bets on other outcomes are considered as lost bets. The simulations with this strategy are displayed in Figure 4.5. The variance-adjusted strategy that goes all in on new predictions also gave a positive average yield. Compared to the betting simulation where we only placed one bet each match, the difference between the minimum and maximum yields are much smaller. However, the increased number of bets means that the yield has a greater impact on the final bankroll value. A yield at -5.43, which is the case with the worst run using the Kelly strategy, results in a bankrupt simulation. This is shown in Figure 4.6.

Table 4.6: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-5.43	-1.43	-4.76	-1.97
Maximum	0.88	2.57	1.01	2.19
Average	-1.53	0.60	-1.19	0.19

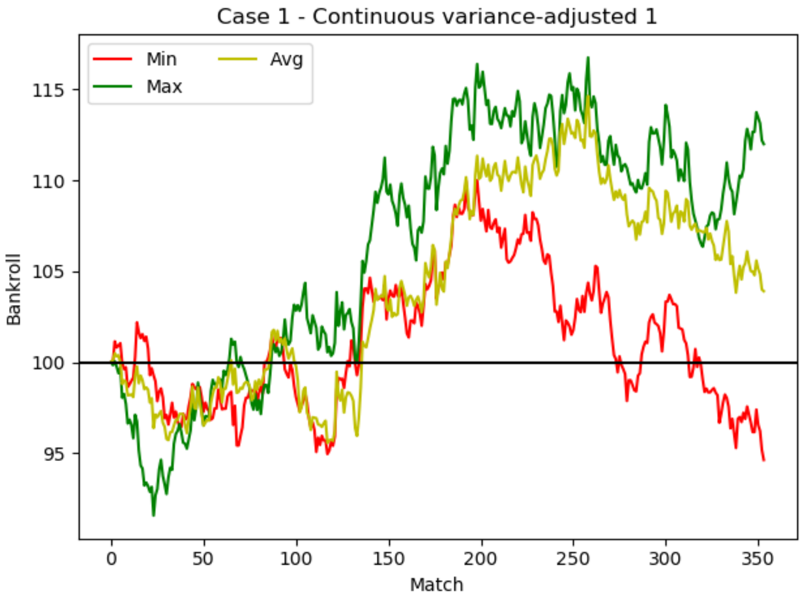


Figure 4.5: Bankroll values with the variance-adjusted strategy where previous bets on other outcomes are considered as lost bets. Shows the worst and the best simulation, and the average bankroll value.

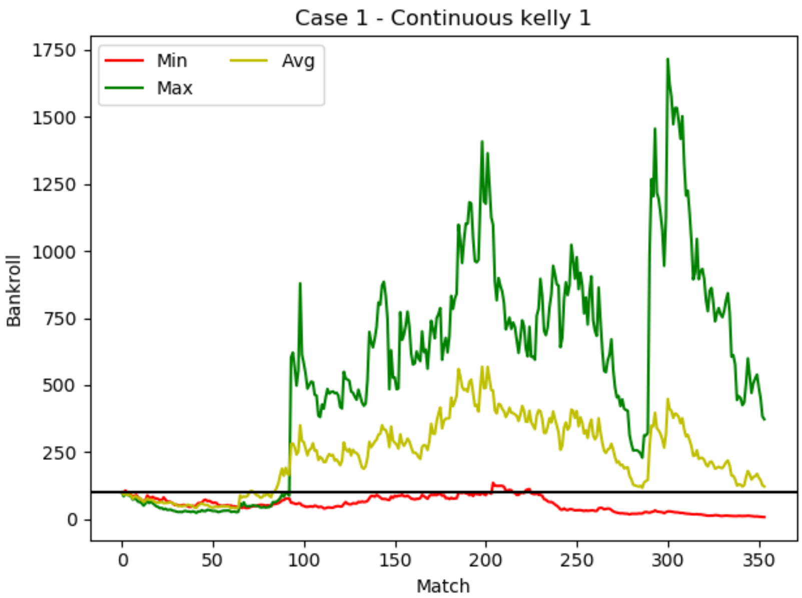


Figure 4.6: Bankroll values with the Kelly strategy where previous bets on other outcomes are considered as lost bets. Shows the worst and the best simulation, and the average bankroll value. The worst simulation goes bankrupt.

These strategies have much better results than when placing multiple bets in each match using the regular Kelly and variance-adjusted strategies. The regular versions of these strategies do not take previous bets into account. This means that if the strategy recommends you to bet 10 units at the 5th minute and the odds and probabilities remain unchanged, it will recommend you to bet 10 new units. Table 4.7 shows the yields using these strategies. Only the variance-adjusted's best run had a positive yield. Using the Kelly strategy, even the best run went bankrupt before 50 matches. This is shown in Figure 4.7.

Table 4.7: Yields, in percent, when possibly placing multiple bets in each match and not considering previous bets.

	Kelly	Variance-adjusted
Minimum	-70.97	-6.99
Maximum	-17.99	1.33
Average	-40.67	-1.72

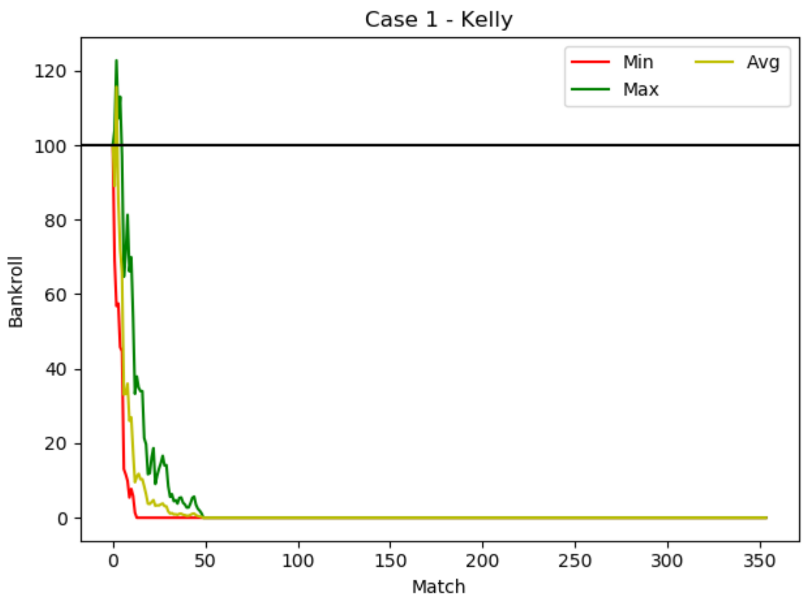


Figure 4.7: Bankroll values with the regular Kelly strategy. Shows the worst and the best simulation, and the average bankroll value. All simulation goes bankrupt.

4.3.3 Case 2

Case 2 consists of data from all the leagues, and uses only EGRT and xG as input.

Network parameters

Table 4.8 shows the ten combinations of network parameters that obtained the highest accuracies, for both the feedforward network and the RNN. A feedforward network with 2 hidden layers, 4 nodes in each layer, the tanh activation function, a 0.002 learning rate, and batch size of 64 gave the highest validation accuracy. This network had a validation accuracy of 0.55002, which is higher than the best network in Case 1. In fact, all the top ten combinations of network parameters achieved a validation accuracy better than the top network in Case 1, and also better than the baselines. Compared to Case 1, the highest validation accuracies were achieved with feedforward networks in Case 2.

Table 4.8: Best network parameters with Case 2. The blue-colored background indicates an RNN.

Number of hidden layer(s) and nodes in each layer	Hidden layer(s) activation	Learning rate	Batch size	Validation accuracy
2 layers 4 nodes	tanh	0.002	64	**0.55002**
2 layers 5 nodes	relu	0.002	32	**0.54709**
1 layers 5 nodes	sigmoid	0.002	128	**0.54666**
2 layer 5 nodes	tanh	0.002	256	**0.54531**
2 layer 4 nodes	tanh	0.002	128	**0.54463**
1 layer 5 nodes	sigmoid	0.002	64	**0.54417**
2 layers 64 nodes	LSTM with tanh	0.001	-	**0.54400**
1 layer 5 nodes	tanh	0.0005	32	**0.54374**
2 layers 5 nodes	tanh	0.002	32	**0.54349**
2 layer 5 nodes	tanh	0.0005	256	**0.54328**

Betting simulation 1

Figures 4.8 to 4.11 show how the bankroll develops during the betting simulations, using the different betting strategies while betting at random time points. The yields from the simulation are showed in Table 4.9. For Case 2, using the variance-adjusted strategy gave the best average yield. However, this is still not better than the baseline.

Table 4.9: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-11.86	-8.03	-12.35	-5.42
Maximum	5.25	6.95	-3.59	7.52
Average	-2.84	0.14	-7.25	1.91

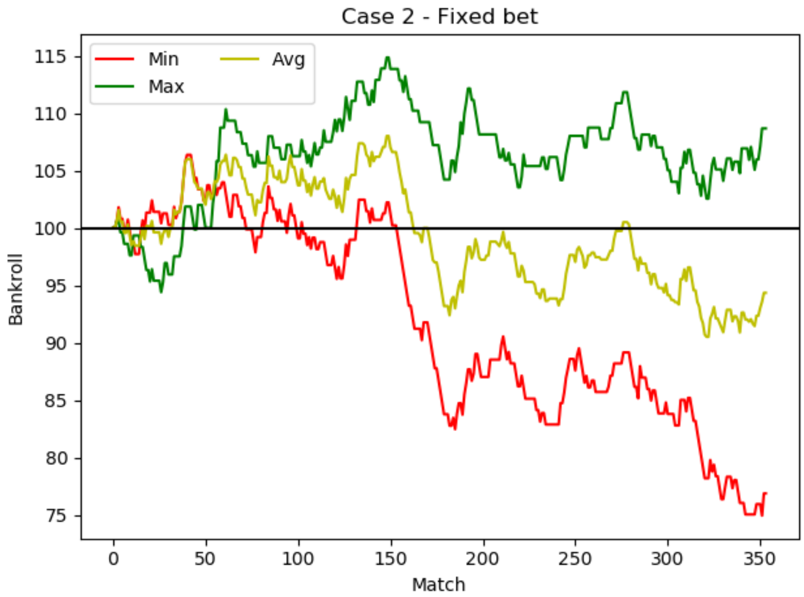


Figure 4.8: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

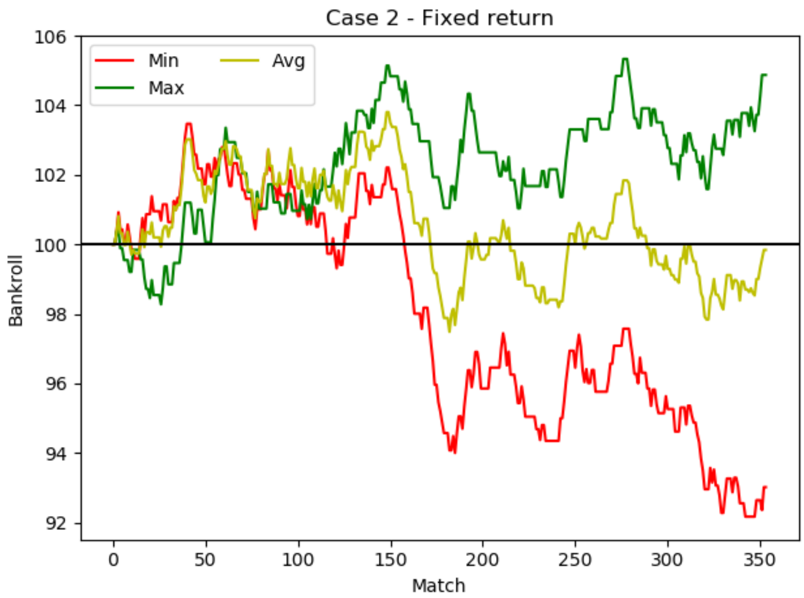


Figure 4.9: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

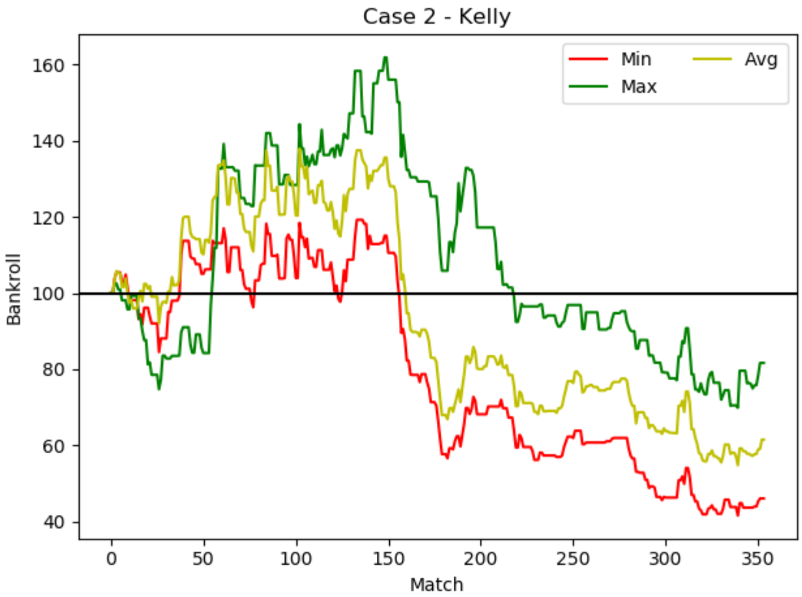


Figure 4.10: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

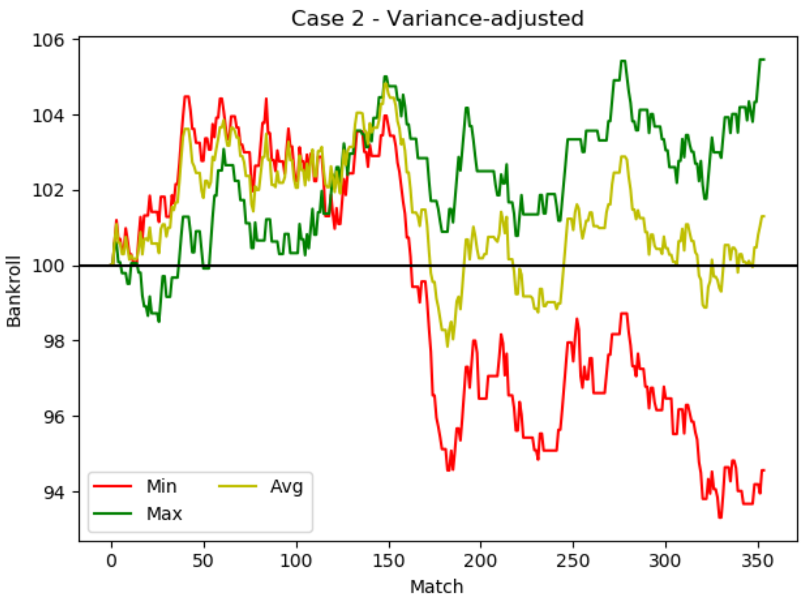


Figure 4.11: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

We can observe from the figures that the gap between the worst and best run is big in Case 2, as it was with Case 1. The Kelly strategy graphs show that the runs are pretty similar, which also can be read from the final yield values. However, even the best run gave a negative Yield using the Kelly strategy. Two of the strategies, the fixed return and variance-adjusted, performed better than Case 1 on average, but the worst run was better in Case 1.

Betting simulation 2

Table 4.10 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. Similar to Case 1, placing bets late in the game gave positive yields. The highest yield was achieved using the fixed return strategy, at the 90th minute in the matches. This gave a yield of 34.96, which is better than the baseline.

Table 4.10: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	-9.54	-9.18	-9.42	-8.47
30	-8.29	-6.36	-11.08	-5.32
45	-11.97	-9.80	-18.48	-8.25
60	-13.16	-11.55	-16.67	-10.97
75	5.13	4.97	-2.48	4.23
90	26.03	34.96	-9.25	34.77

Betting simulation 3

Table 4.11 shows the results from the simulations where multiple bets could be placed in each match. None of the strategies generated a positive yield, not even in their best run. The best average yield was achieved using the variance-adjusted strategy that goes all in on the new prediction. The Kelly strategies resulted in bankruptcy, also in their best runs. Figure 4.12 shows the simulations using the Kelly strategy that goes all in on the new prediction.

Table 4.11: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-8.80	-2.55	-7.53	-2.62
Maximum	-5.50	-1.03	-4.91	-0.83
Average	-7.51	-1.61	-6.53	-1.50

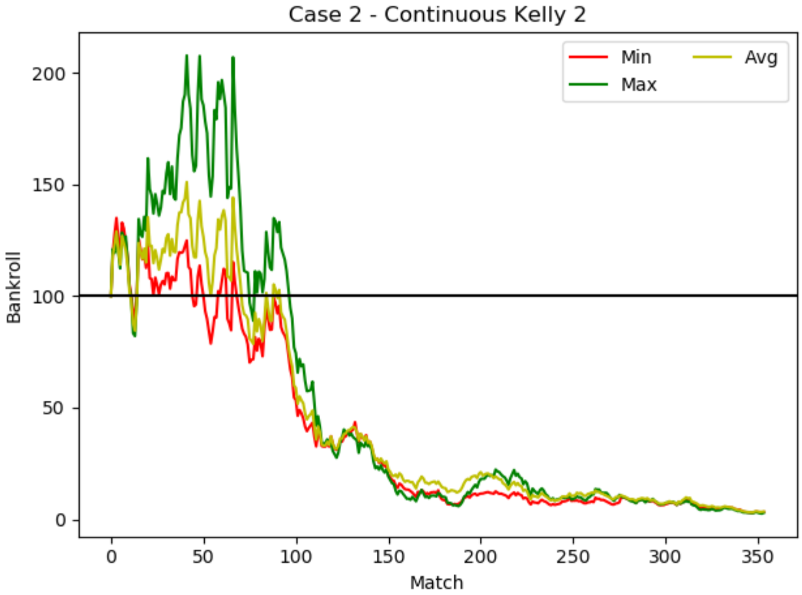


Figure 4.12: Bankroll values with the Kelly strategy that goes all in on new predictions. Shows the worst and the best simulation, and the average bankroll value. All goes bankrupt.

4.3.4 Case 3

Case 3 consists of data from all the leagues and uses match events as input.

Network parameters

Table 4.12 shows the ten combinations of network parameters that obtained the highest accuracies. Unlike the two previous cases, all the top accuracies were achieved by the feedforward network. A feedforward network with 2 hidden layers, 53 nodes in each layer, the tanh activation function, a 0.001 learning rate, and batch size of 256 gave the highest validation accuracy. With 0.51401, this accuracy is lower than the ones achieved by the previous cases but still better than the baselines.

Betting simulation 1

Figures 4.13 to 4.16 show how the bankroll develops during the betting simulations, using the different betting strategies while betting at random time points. The yields from the simulation are shown in Table 4.13. For Case 3, using the variance-adjusted strategy gave the best average yield, which is also significantly better than than the baseline. Additionally, the fixed bet and fixed return strategies had an average yield better than the baseline.

Table 4.12: Best network parameters with Case 3.

Number of hidden layer(s) and number of nodes in each layer	**Hidden layer(s) activation**	**Learning rate**	**Batch size**	**Validation accuracy**
2 layers 53 nodes	tanh	0.001	256	**0.51401**
2 layers 53 nodes	tanh	0.001	32	**0.51104**
1 layer 53 nodes	tanh	0.001	256	**0.50997**
2 layers 28 nodes	tanh	0.001	128	**0.50997**
1 layer 53 nodes	sigmoid	0.002	32	**0.50952**
2 layers 28 nodes	relu	0.002	64	**0.50909**
2 layers 53 nodes	tanh	0.0005	128	**0.50906**
1 layer 28 nodes	relu	0.0005	32	**0.50906**
1 layer 28 nodes	sigmoid	0.001	512	**0.50885**
1 layer 28 nodes	tanh	0.001	512	**0.50885**

Table 4.13: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	**Fixed return**	**Kelly**	**Variance-adjusted**
Minimum	-0.58	4.45	-6.31	8.99
Maximum	10.66	15.65	0.12	18.31
Average	6.37	11.87	-1.68	14.57

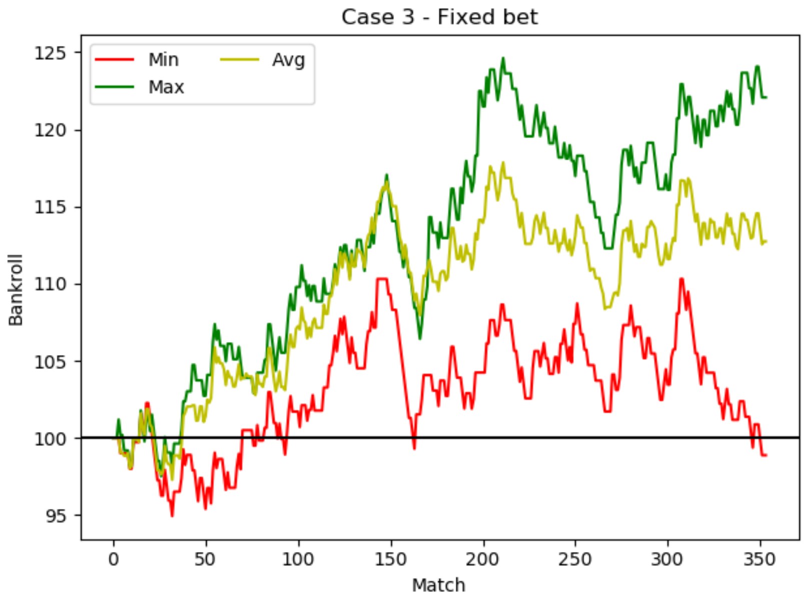


Figure 4.13: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

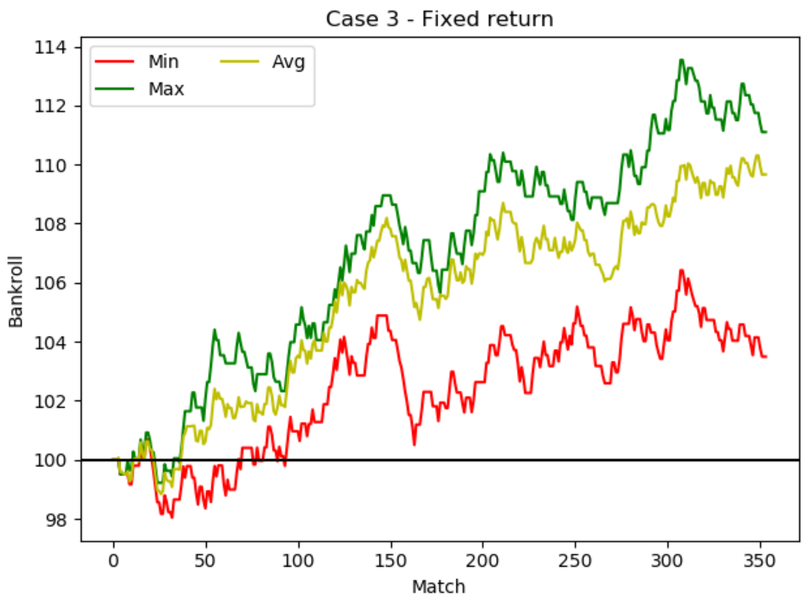


Figure 4.14: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

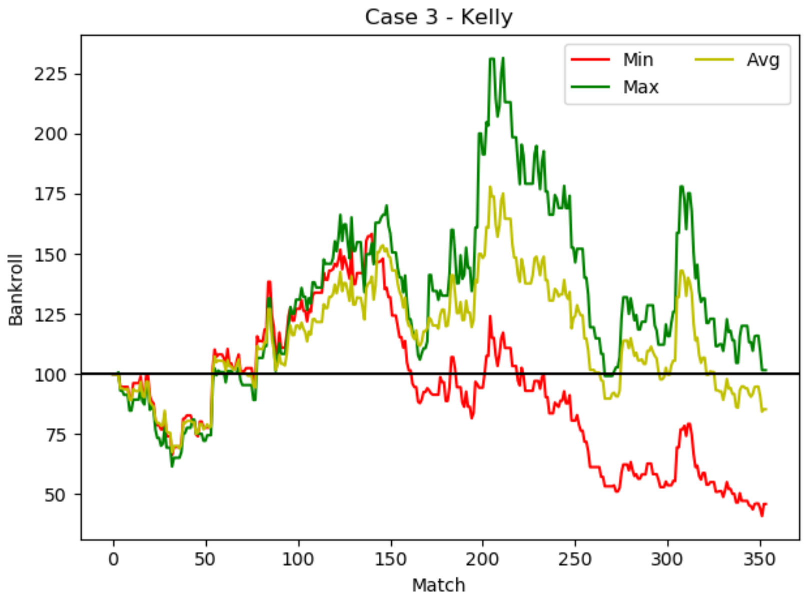


Figure 4.15: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

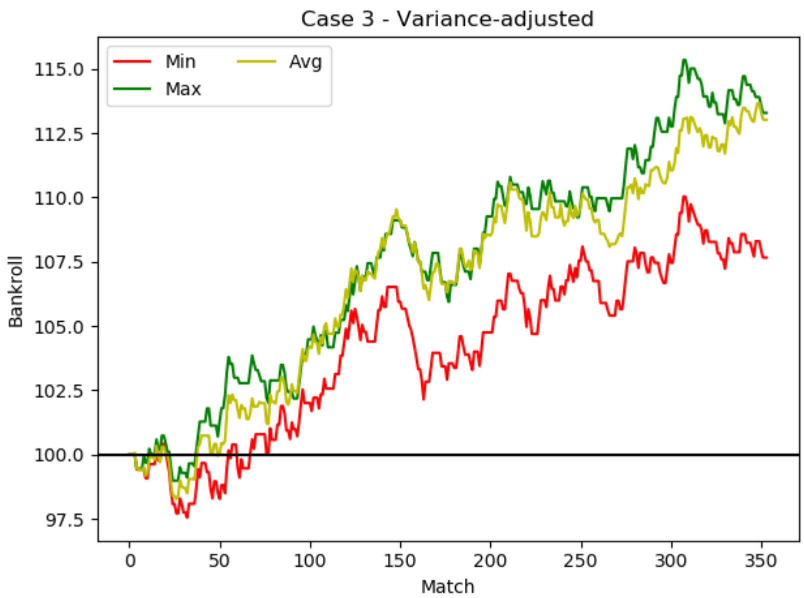


Figure 4.16: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

As shown in the figures, the difference between the worst and the best runs are smaller, compared to the previous cases. Two of the strategies, the variance-adjusted and fixed return strategy, also achieved a positive yield on their worst run. The Kelly strategy's results are still rather poor, where the best run is barely giving a positive yield.

Betting simulation 2

Table 4.14 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. Placing bets at the 75th and 90th gives a positive yield for all the strategies, except for the Kelly strategy. The Kelly strategy gives a negative yield for all the time points, which matches the results from the first betting simulation. The three other strategies also give a positive yield at the 30th minute, which has not been the case for the previous cases. The highest yield was achieved using the variance-adjusted strategy, at the 90th minute in the matches.

Table 4.14: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	-1.98	-0.89	-4.86	0.43
30	2.28	3.58	-3.55	3.86
45	-5.67	-1.01	-11.44	0.24
60	-1.97	-3.28	-4.91	-3.32
75	2.37	3.25	-3.20	3.36
90	8.09	27.09	-43.46	28.06

Betting simulation 3

Table 4.15 shows the results from the simulations where multiple bets could be placed in each match. Considering the good results from the first betting simulation, the results from this simulation were rather poor. None of the strategies produced a positive average yield. The best average yield was achieved using the variance-adjusted strategy that goes all in on the new prediction. The simulations with this strategy are displayed in Figure 4.17.

Table 4.15: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-16.52	-3.55	-13.88	-3.24
Maximum	-5.51	0.56	-4.58	0.45
Average	-9.52	-0.63	-8.59	-0.49

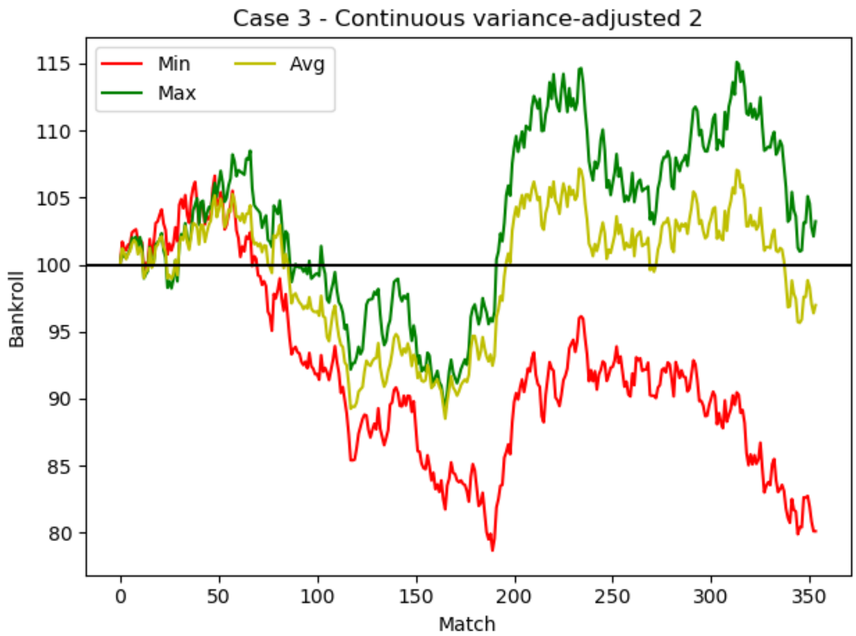


Figure 4.17: Bankroll values with the variance-adjusted strategy that goes all in on the new prediction. Shows the worst and the best simulation, and the average bankroll value.

4.3.5 Case 4

Case 4 consists of data from the English Premier League and uses all the available features as input.

Network parameters

Table 4.16 shows the ten combinations of network parameters that obtained the highest accuracies. All the top accuracies were achieved by the RNN. An RNN with 2 LSTM layers, 64 units in each layer, and a 0.001 learning rate gave the highest validation accuracy. This setup achieved an accuracy of 0.56274, which is better than the previous cases.

Betting simulation 1

Figures 4.18 to 4.21 show how the bankroll develops during the betting simulations, using the different betting strategies while betting at random time points. The yields from the simulation are showed in Table 4.17. For Case 4, using the Kelly strategy gave the best average yield, which was not the case for any of the previous cases. The Kelly strategy's average yield is also better than the baseline. The other strategies gave a positive average yield as well.

Table 4.16: Best network parameters with Case 4. The blue-colored background indicates an RNN.

Number of hidden layer(s) and number of nodes in each layer	Hidden layer(s) activation	Learning rate	Batch size	Validation accuracy
2 layers 64 nodes	LSTM with tanh	0.001	-	**0.56274**
2 layers 64 nodes	LSTM with tanh	0.003	-	**0.55283**
4 layers 64 nodes	LSTM with tanh	0.001	-	**0.55221**
6 layers 64 nodes	LSTM with tanh	0.001	-	**0.54660**
4 layers 128 nodes	LSTM with tanh	0.001	-	**0.54512**
4 layers 32 nodes	LSTM with tanh	0.001	-	**0.54431**
2 layers 128 nodes	LSMT with tanh	0.003	-	**0.54319**
2 layers 64 nodes	LSTM with tanh	0.002	-	**0.54290**
2 layers 32 nodes	LSTM with tanh	0.001	-	**0.54285**
2 layers 32 nodes	LSTM with tanh	0.003	-	**0.54249**

Table 4.17: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-1.73	-6.65	-6.83	-7.30
Maximum	8.24	8.29	17.51	15.72
Average	2.60	2.90	6.98	3.81

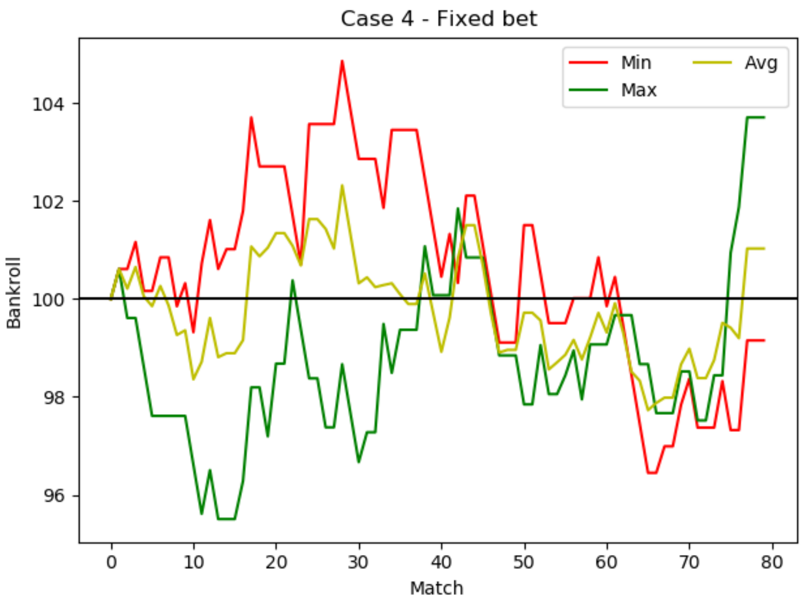


Figure 4.18: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

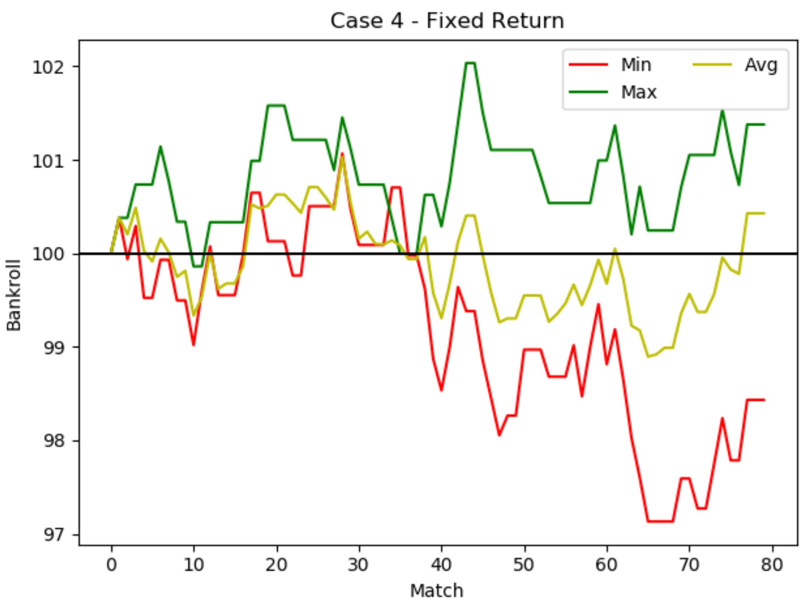


Figure 4.19: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

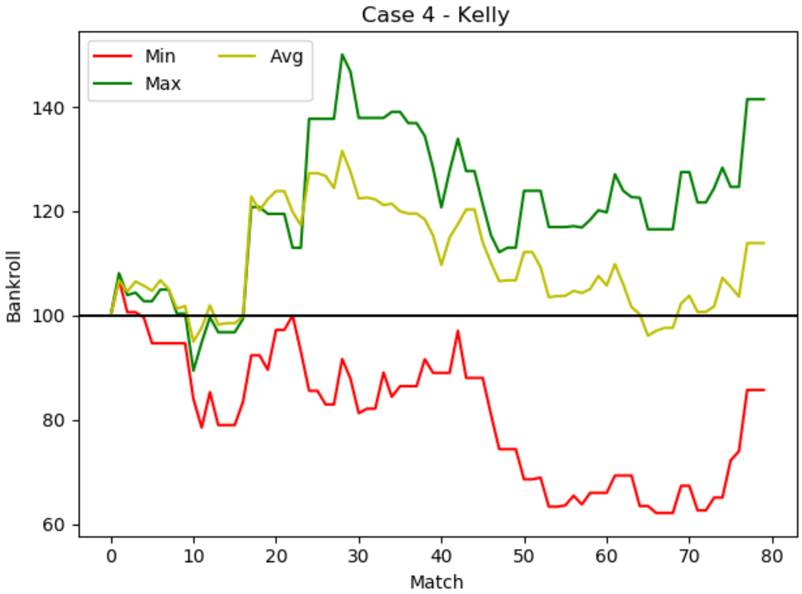


Figure 4.20: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

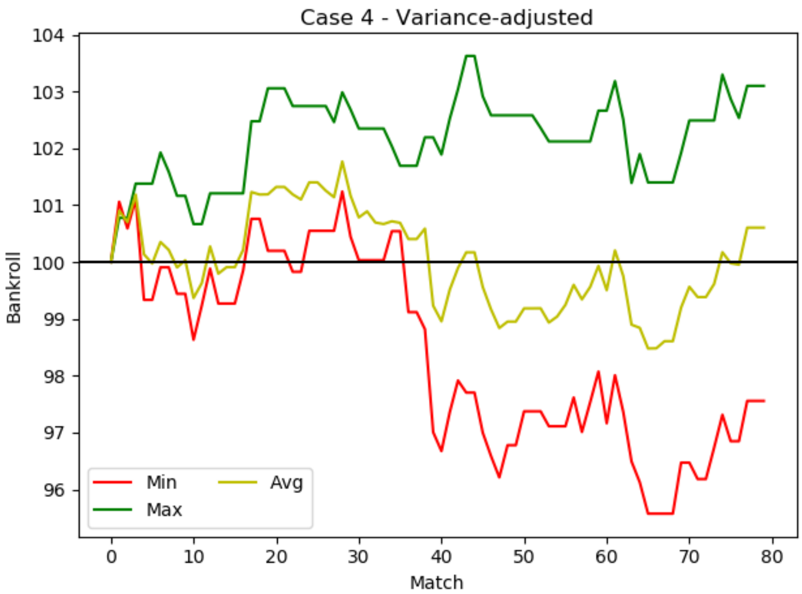


Figure 4.21: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

The figures show that the worst run generates a negative yield with all the strategies. There is a big difference between the worst and best run, indicating that the network is not very robust. However, the test set is much smaller for Case 4, as it only includes matches from the Premier League. Therefore, each match has a greater impact on the final yield. This is the only case that generated a positive average yield for all the strategies, which we can see from the yellow graphs. The best runs also gave high yield for all the strategies, where the previous cases had bigger differences between the strategies.

Betting simulation 2

Table 4.18 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. Unlike the previous cases, Case 4 gave positive yields when placing bets at the 15th and 30th minute, with all the strategies. Three of the strategies also gave positive yields at the 60th minute, except Kelly. Similar to the other cases, placing bets at the 90th minute gave high yield. The highest yield was achieved using the Kelly strategy, at the 90th minute in the matches.

Table 4.18: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	8.15	5.69	16.37	6.29
30	5.32	2.27	12.81	4.29
45	-4.51	-4.33	-8.74	-1.74
60	7.60	5.66	-11.54	2.65
75	-2.80	-1.72	-10.23	-2.36
90	46.51	46.46	47.84	46.56

Betting simulation 3

Table 4.19 shows the results from the simulations where multiple bets could be placed in each match. All the strategies gave a positive average yield, where both Kelly strategies gave yields that are better than the baselines. The best average yield was achieved using the Kelly strategy that goes all in on the new prediction. The simulations with the best strategy are displayed in Figure 4.22. This strategy gave an average yield of 11.72, and a maximum Yield of 23.91. As seen from the charts, the simulation giving the maximum yield ended with finale bankroll value over 1750, while the finale bankroll value is slightly above 900 on average.

Table 4.19: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-2.71	-7.77	-1.71	-5.17
Maximum	23.96	8.87	23.91	8.26
Average	10.59	2.68	11.72	2.75

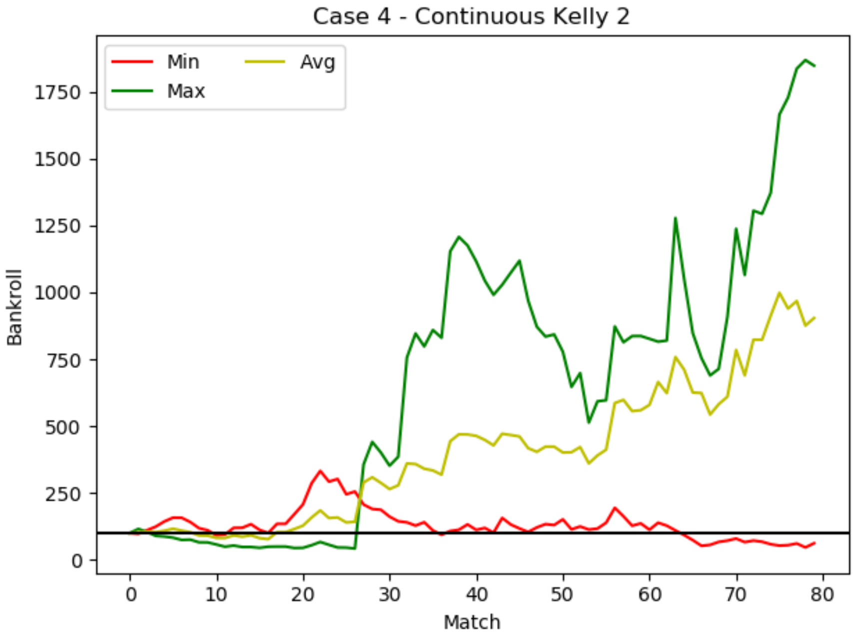


Figure 4.22: Bankroll values with the Kelly strategy that goes all in on the new prediction. Shows the worst and the best simulation, and the average bankroll value.

4.3.6 Case 5

Case 5 consists of data from the English Premier League and uses only EGRT and xG as input.

Network parameters

Table 4.20 shows the ten combinations of network parameters that obtained the highest accuracies. For Case 5, all the top accuracies were achieved by the feedforward network. A feedforward network with 1 hidden layer, 5 nodes in the hidden layer, the tanh activation function, a 0.001 learning rate, and batch size of 64 gave the highest validation accuracy. This gave an accuracy of 0.60052, which is by far the highest accuracy achieved out of all the cases. All top ten combinations of parameters for Case 5 gave a higher validation accuracy than what has been achieved with the other cases.

Betting simulation 1

Figures 4.23 to 4.26 show how the bankroll develops during the betting simulations, using the different betting strategies while betting at random time points. The yields from the simulation are showed in Table 4.21. For Case 5, using the variance-adjusted strategy gave the best average yield. Further, all strategies generated a positive yield on average, but none were better than the top baseline.

Table 4.20: Best network parameters with Case 5.

Number of hidden layer(s) and number of nodes in each layer	Hidden layer(s) activation	Learning rate	Batch size	Validation accuracy
1 layer 5 nodes	tanh	0.001	64	**0.60052**
1 layer 4 nodes	relu	0.002	32	**0.60050**
1 layer 5 nodes	tanh	0.002	32	**0.59813**
1 layer 5 nodes	tanh	0.002	256	**0.59577**
1 layer 4 nodes	tanh	0.001	64	**0.59460**
1 layer 4 nodes	tanh	0.002	64	**0.59220**
1 layer 4 nodes	relu	0.0005	32	**0.58980**
1 layer 5 nodes	sigmoid	0.002	32	**0.58868**
1 layer 5 nodes	tanh	0.002	64	**0.58863**
1 layer 4 nodes	tanh	0.001	32	**0.58746**

Table 4.21: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-25.94	-19.83	-28.50	-15.15
Maximum	31.82	22.36	25.32	21.83
Average	2.02	1.27	0.19	3.84

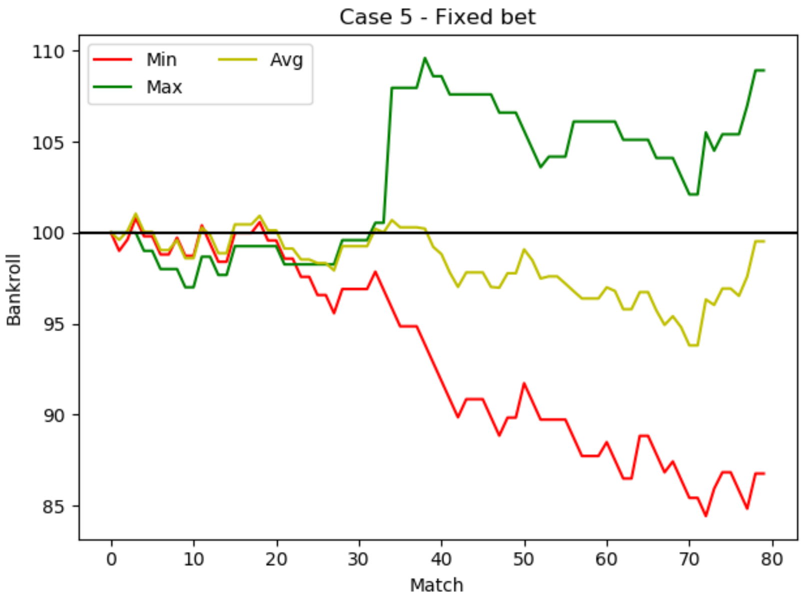


Figure 4.23: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

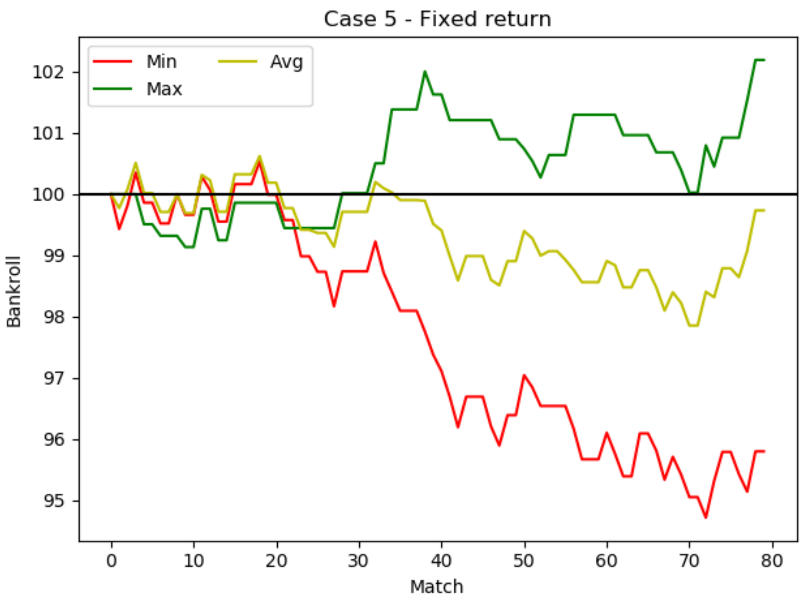


Figure 4.24: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

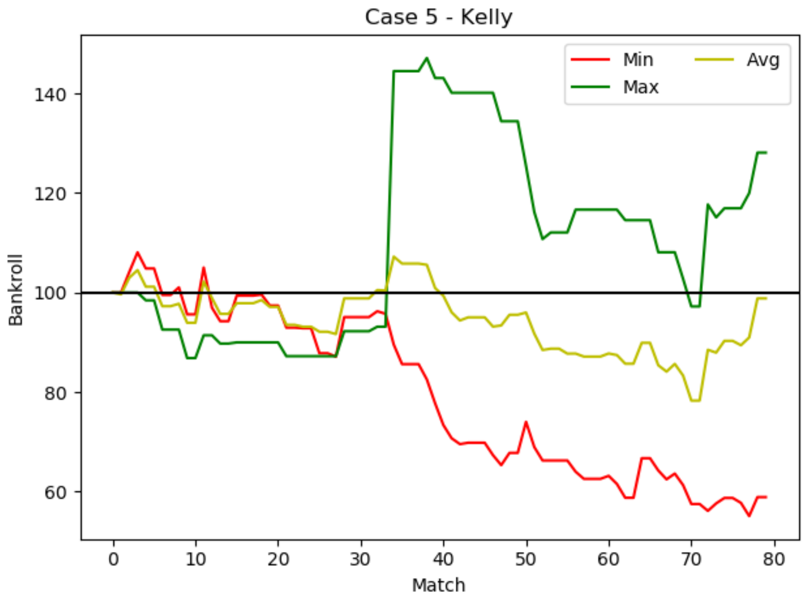


Figure 4.25: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

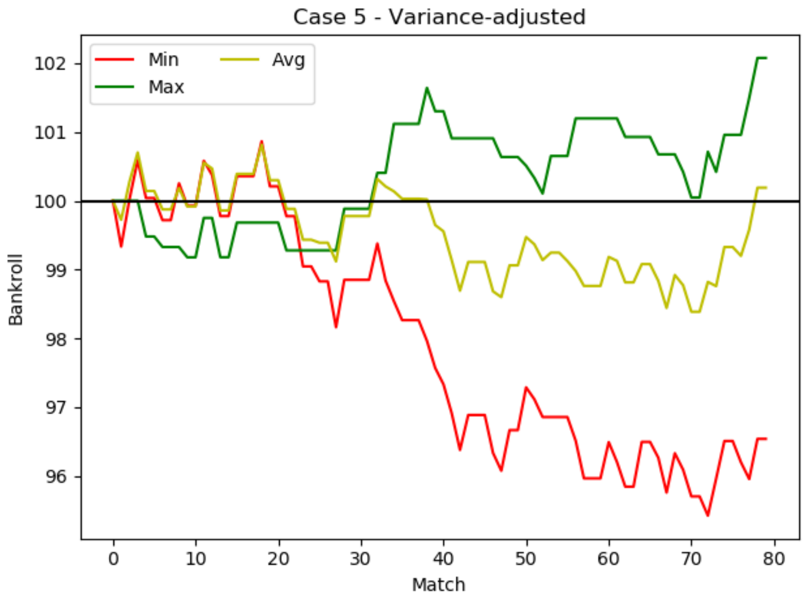


Figure 4.26: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

As with the previous case, we can see a big difference between the best and the worst run. Common for all the strategies is a large positive yield for the best runs and a similarly large negative yield for the worst runs. The yellow graph reveals that all of the strategies, on average, finished with a bankroll bigger than the initial bankroll, which is also visible from the yields in Table 4.21.

Betting simulation 2

Table 4.22 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. As with Case 4, Case 5 also generated positive yields when placing bets at the 15th and 30th minute. The only exception is using the fixed return strategy at the 15th minute, which gave a yield of -0.57. A negative yield was achieved when placing bets at the other time points. All the other time points gave negative yields, with the exception of the Kelly strategy at the 45th minute. At the 90th minute, none of the strategies placed any bets in any of the 79 matches. The highest yield was achieved using the Kelly strategy, at the 30th minute in the matches.

Table 4.22: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	0.08	-0.57	4.07	1.74
30	10.71	7.79	12.90	8.96
45	-5.77	-10.43	12.38	-7.01
60	-10.80	-11.64	-23.53	-12.74
75	-2.05	-1.59	-10.75	-1.82
90	No bets taken	No bets taken	No bets taken	No bets taken

Betting simulation 3

Table 4.23 shows the results from the simulations where multiple bets could be placed in each match. All strategies gave a negative average yield, and only the variance-adjusted strategies gave a positive yield on their best run. The best average yield was achieved using the variance-adjusted strategy that goes all in on the new prediction. The simulations with this strategy are displayed in Figure 4.27.

Table 4.23: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-8.56	-5.56	-6.80	-2.79
Maximum	-1.75	0.32	-1.41	0.59
Average	-6.01	-1.27	-4.14	-0.45

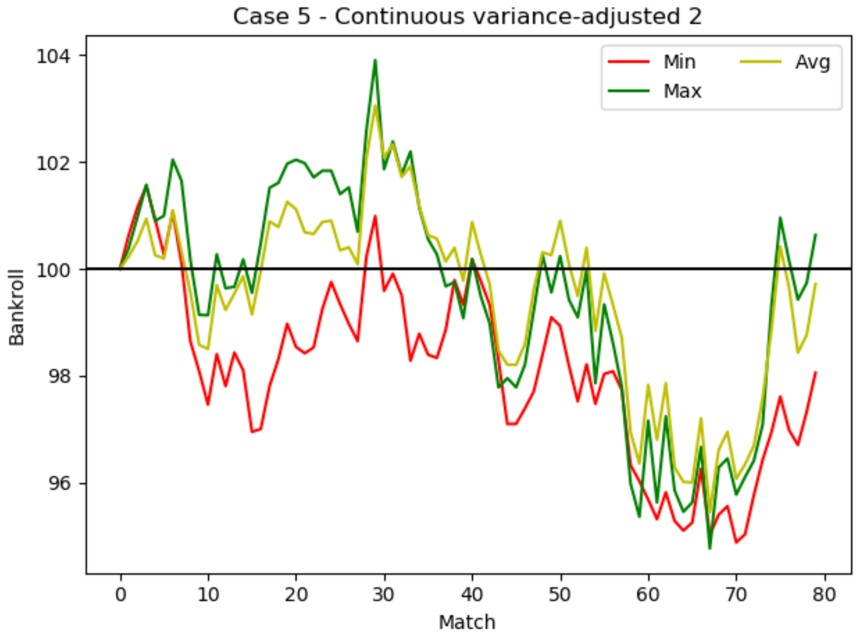


Figure 4.27: Bankroll values with the variance-adjusted strategy where previous bets on other outcomes are considered as lost bets. Shows the worst and the best simulation, and the average bankroll value.

4.3.7 Case 6

Case 6 consists of data from the English Premier League and uses match events as input.

Network parameters

Table 4.24 shows the ten combinations of network parameters that obtained the highest accuracies. All the top accuracies were achieved by the feedforward network while betting at random time points. A feedforward network with 1 hidden layer, 53 nodes in the hidden layer, the tanh activation function, a 0.0005 learning rate, and a batch size of 64 gave the highest validation accuracy. This setup gave an accuracy of 0.53754, which is higher than the top accuracy in Case 3, which also only used match events as input.

Betting simulation 1

Figures 4.28 to 4.31 show how the bankroll develops during the betting simulations, using the different betting strategies. The yields from the simulation are showed in Table 4.25. For Case 6, using the variance-adjusted strategy gave the best average yield. This was the only strategy that achieved a positive average yield, which also is better than the top baseline.

Table 4.24: Best network parameters with Case 6.

Number of hidden layer(s) and number of nodes in each layer	Hidden layer(s) activation	Learning rate	Batch size	Validation accuracy
1 layer 53 nodes	tanh	0.0005	64	**0.53754**
2 layers 53 nodes	relu	0.002	64	**0.53635**
2 layers 28 nodes	tanh	0.001	32	**0.53630**
1 layer 28 nodes	relu	0.002	32	**0.53512**
1 layer 28 nodes	tanh	0.002	256	**0.53397**
2 layers 53 nodes	tanh	0.002	64	**0.53394**
3 layers 28 nodes	tanh	0.001	32	**0.53273**
1 layer 28 nodes	tanh	0.0005	64	**0.53036**
1 layer 28 nodes	relu	0.002	256	**0.53034**
1 layer 53 nodes	tanh	0.001	256	**0.53034**

Table 4.25: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-7.47	-7.13	-8.25	1.92
Maximum	3.66	7.88	3.69	14.72
Average	-1.77	0.81	-1.94	9.33

Figure 4.28: Bankroll values with the fixed bet strategy. Shows the worst and the best simulation, and the average bankroll value.

Figure 4.29: Bankroll values with the fixed return strategy. Shows the worst and the best simulation, and the average bankroll value.

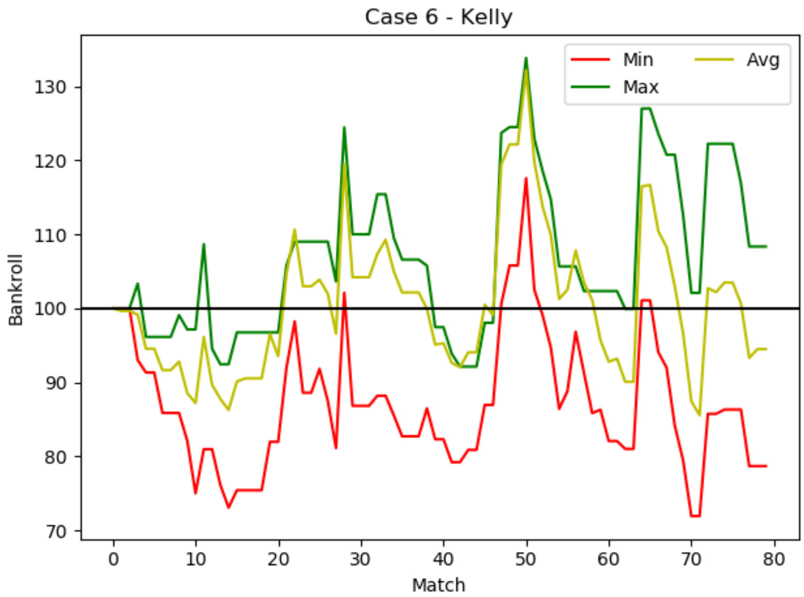


Figure 4.30: Bankroll values with the Kelly strategy. Shows the worst and the best simulation, and the average bankroll value.

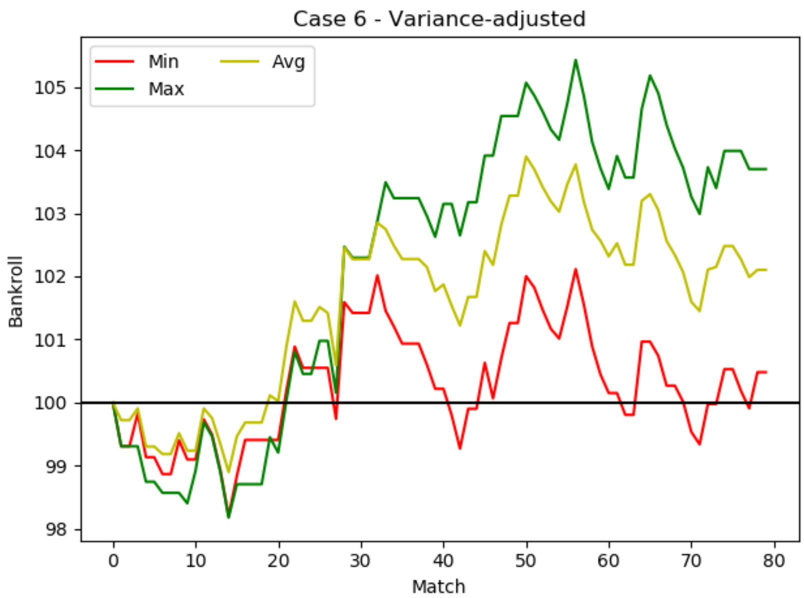


Figure 4.31: Bankroll values with the variance-adjusted strategy. Shows the worst and the best simulation, and the average bankroll value.

Similar to the two other cases using only matches from the Premier League, the difference between the best and worst run is big. This can be seen in the figure. However, the variance-adjusted strategy gives a positive yield even on its worst run, which is shown with the red graph.

Betting simulation 2

Table 4.26 shows the yields from the simulations where bets were placed at the same fixed time point in each simulation. A positive yield was achieved when placing bets at the 15th, 30th and 60th minute, with all the strategies. The other time points generated negative yields, with all strategies. The highest yield was achieved using the variance-adjusted strategy, at the 15th minute in the matches.

Table 4.26: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	10.75	11.43	16.16	19.83
30	5.36	7.49	5.45	13.72
45	-7.10	-6.38	-14.64	-5.15
60	6.67	1.23	5.98	0.22
75	-15.36	-16.05	-8.52	-15.35
90	-10.25	-10.25	-10.25	-10.25

Betting simulation 3

Table 4.27 presents the results from the simulations where multiple bets could be placed in each match. All the strategies produced a positive average yield, but none that were better than the best baseline. Only the best Kelly run, where previous bets on other outcomes are considered as lost bets, achieved a yield better than the best baseline. Further, the variance-adjusted strategy where previous bets on other outcomes are considered as lost bets finished with positive yield on its worst run. This strategy also achieved the best average yield. The simulations with this strategy are displayed in Figure 4.32.

Table 4.27: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-2.42	0.82	-2.73	-0.20
Maximum	6.37	4.01	5.52	4.76
Average	1.87	2.44	1.53	2.18

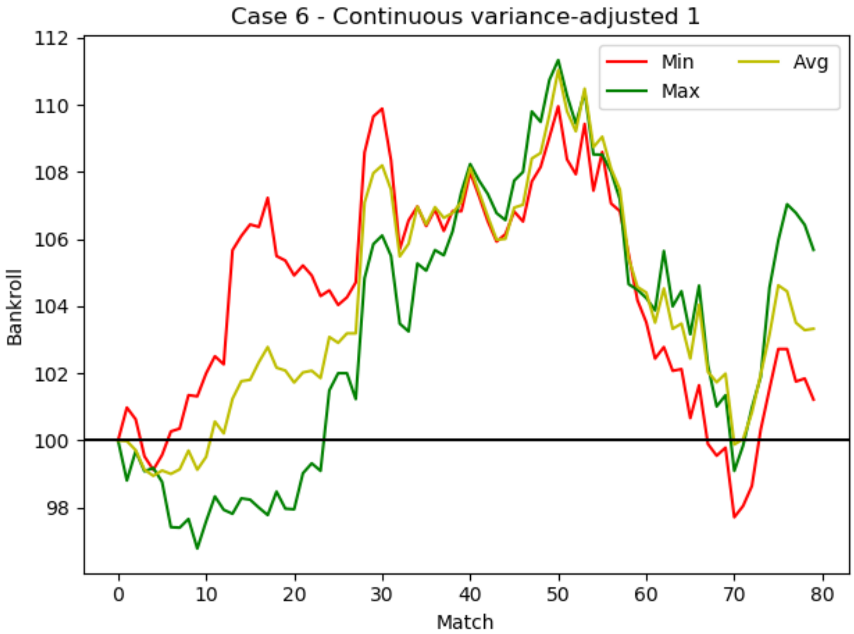


Figure 4.32: Bankroll values with the variance-adjusted strategy where previous bets on other outcomes are considered as lost bets. Shows the worst and the best simulation, and the average bankroll value.

4.3.8 Summary

The first part of experiments, finding network parameters that resulted in a high validation accuracy, delivered promising results for all the cases. All cases had a validation accuracy, with their top setup, higher than the baseline at 0.44. Case 5 achieved the highest accuracy, at 0.60052.

Table 4.28: The average yield values, where one bet was placed in each match at a random time. Yellow-colored background marks a positive average yield. Green-colored background indicates that also the worst run had a positive yield.

	Fixed bet	**Fixed return**	**Kelly**	**Variance-adjusted**
Case 1	-0.15	-0.24	-4.43	1.11
Case 2	-2.84	0.14	-7.25	1.91
Case 3	6.37	11.87	-1.68	14.57
Case 4	2.60	2.90	6.98	3.81
Case 5	2.02	1.27	0.19	3.84
Case 6	-1.77	0.81	-1.94	9.33

Table 4.28 summarizes the yield achieved by all the betting strategies in the different cases for the first betting simulation. For the first betting simulation, where only one bet was placed in each match, all cases were able to generate profit with at least one of the strategies. The

variance-adjusted strategy achieved a positive yield with all cases, while the other strategies had varying results. The Kelly strategy performed the worst, and only achieved a positive yield in Case 4 and Case 5. This strategy is also the only strategy that went bankrupt in some runs. However, in Case 4, the Kelly strategy achieved the highest average yield out of all the strategies. Case 4 and Case 5 were also the only cases that achieved positive average yields with all betting strategies.

There are three combinations of cases and betting strategies that achieved a positive yield also on their worst run. Out of these, the one using the variance-adjusted strategy with Case 3 gave the highest average yield at 14.57. This combination's worst run had a yield as big as 8.99.

For the second part of the simulations, where bets were placed at fixed time points, the results varied. In general, a positive yield was achieved when betting late in the match with the three first cases, which consisted of data from all five leagues. For the three last cases, which consisted of data from only the Premier League, a positive yield was achieved when betting early in the match. The highest yield, 54.11, was achieved in Case 1, with the Kelly strategy at the 90th minute.

Table 4.29 summarizes the yield achieved by all the betting strategies in the different cases for the last betting simulation. The last betting simulation allowed multiple bets in each match, where previous bets were taken into consideration. Case 4 and Case 6 achieved positive yield with all the strategies. The variance-adjusted strategy that considered previous bets on other outcomes as lost bets also had a positive yield on its worst in Case 6. Case 2, Case 3 and Case 5 had a negative yield with all strategies.

Table 4.29: The average yield values, where multiple bets were placed in each match. A yellow-colored background marks a positive average yield. A green-colored background indicates that also the worst run had a positive yield.

	Continuous Kelly 1	**Continuous variance-adjusted 1**	**Continuous Kelly 2**	**Continuous variance-adjusted 2**
Case 1	-1.53	0.60	-1.19	0.19
Case 2	-7.51	-1.61	-6.53	-1.50
Case 3	-9.52	-0.63	-8.59	-0.49
Case 4	10.59	2.68	11.72	2.75
Case 5	-6.01	-1.26	-4.14	-0.45
Case 6	1.87	2.44	1.53	2.18

Chapter 5

Discussion

In this chapter, we will discuss the model and take a deeper look into the results from the previous chapter.

5.1 Calibration

The networks output predictions as probabilities. In order to measure how accurate the probabilities generated are, a calibration is performed. This method validates, for instance, whether a 60% prediction actually happens 60% of the times and not just 50%. If an outcome has an odds of 2, there is a difference in how much one should consider betting if the probability of the outcome happening is 60% instead of 50%. This method is also used by FiveThirtyEight, a website specializing in forecasts when they evaluate their predictions [Silver, 2019].

We will take a look at the probabilities generated by the network in Case 3 and Case 4. The data in Case 3 consisted of matches from all the five leagues and used only match events as input. This case had good results from Betting simulation 1, where the variance-adjusted strategy had an average yield of 14.57. However, in Betting simulation 3, none of the strategies used generated a positive average yield. Case 4 had positive yields with all strategies in the two simulations and is the only case achieving this.

Figures 5.1 and 5.2 show the calibration plots for the two cases, which compares predictions to the actual outcomes. The calibrations are performed on the test set used for Betting simulation 3, and includes all the possible outcomes. For both plots, all predictions have been rounded to the nearest ten. The missing points in the charts, e.g. for 90% in Case 3, indicates that the network made no predictions where the outcome had a probability between 85% - 94%. Optimally, each point should be on the diagonal. In general, if a point is in the top triangle, above the diagonal, the predictions are underconfident. This means that the predicted probabilities are lower than the actual probabilities, which can lead to missed opportunities in betting. A point placed in the bottom triangle indicates the predictions being overconfident, meaning the predicted probabilities being higher than the actual probabilities. This can lead to a number of suboptimal bets being placed.

The figures indicate that Case 3 is well calibrated. Most of the points are either on the diagonal, or close to it. The only exception is that it is being underconfident at the 0%-predictions. The calibration for Case 4 also seems to be rather good, but are a little overconfident from the 50%-predictions to 80%-predictions compared to Case 3, meaning the probabilities generated from the neural network are slightly too high.

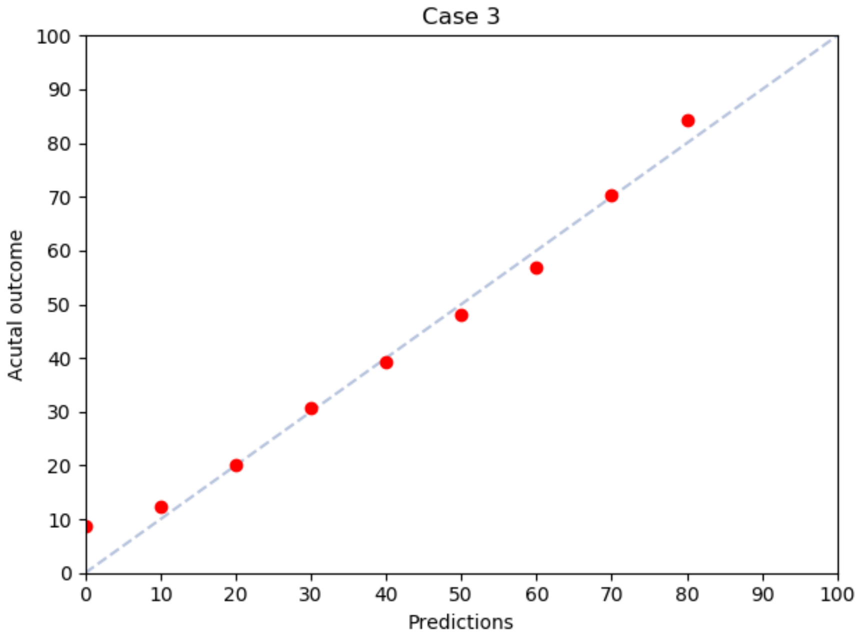


Figure 5.1: Case 3 predictions - Calibration on test set.

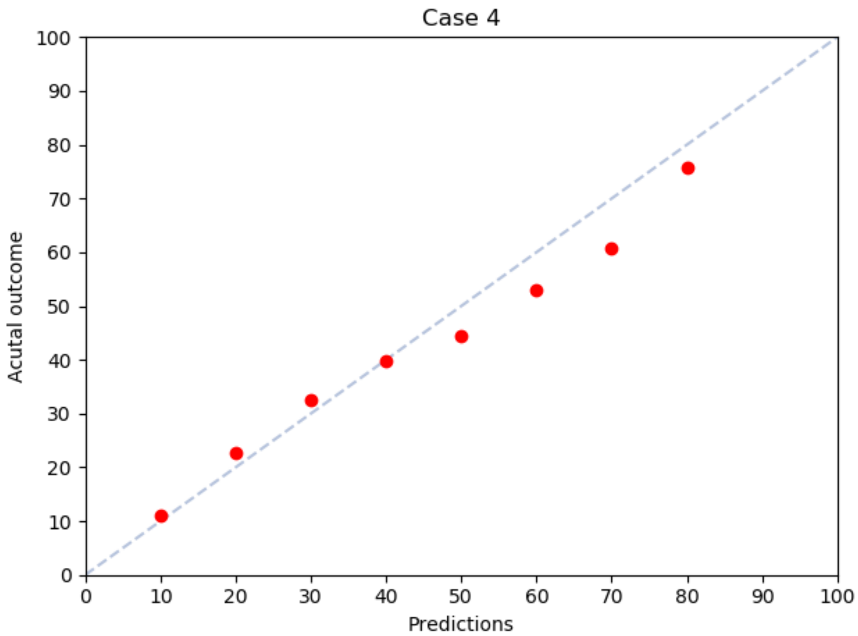


Figure 5.2: Case 4 predictions - Calibration on test set.

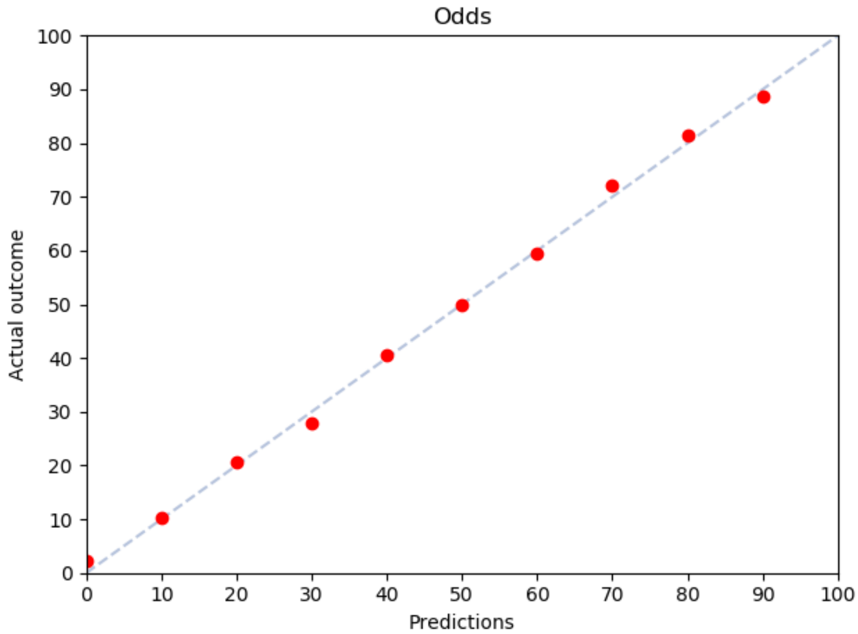


Figure 5.3: Case 3 odds - Calibration on test set.

The results from Case 4, on average, generated positive yields with all betting strategies (see Table 4.28 and 4.29). However, Case 4 had negative yields on the worst run with all strategies. With Case 4 being overconfident, the probabilities made by the neural network are higher than the odds indicate, causing us to bet more than a better-calibrated network. Even if the calibration is rather good, being a little overconfident can ultimately lead to a negative yield. Case 3, being better calibrated than Case 4, produces probabilities that better match the actual outcome. This could be the reason some of the strategies used with Case 3 always have a positive yield, even on the worst run. On the other hand, all the strategies where multiple bets in each match where allowed generated a negative average yield with Case 3.

Further, Figure 5.3 shows the calibration plot for the odds from Sportradar. The figure illustrates that their calibration is pretty good, almost all the points are on the diagonal. As these are the odds the betting simulations are running against, getting an upper hand for a bettor is not easy as they are nearly spot on with their predicted probabilities. However, there are small differences in the probabilities that decides if a bet is to be placed, and we have seen in Section 4.3 that making a profit is possible. Calibration is not everything, and the optimal solution is not a model that is calibrated well. That would be a system that is calibrated and always says that the probability is either 0% or 100%.

5.2 Predictions vs. Actual Outcomes

From the first part of the tests, Case 2 had the worst validation accuracy, while Case 5 achieved the highest. Further, in Betting simulation 1, Case 5 generated a positive average yield with all strategies, while Case 2 had varying results. Figures 5.1 and 5.2 show the confusion matrices generated by the predictions on the test sets, for Case 2 and Case 5 respectively.

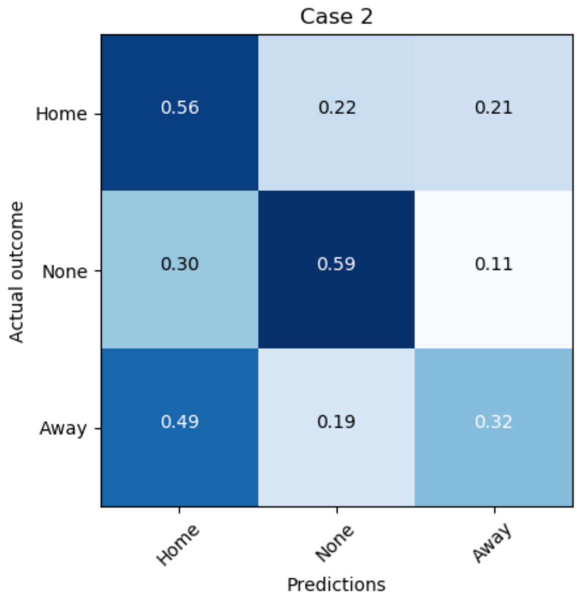


Figure 5.4: Case 2 - Confusion matrix, the normalized row sum is equal to 1.

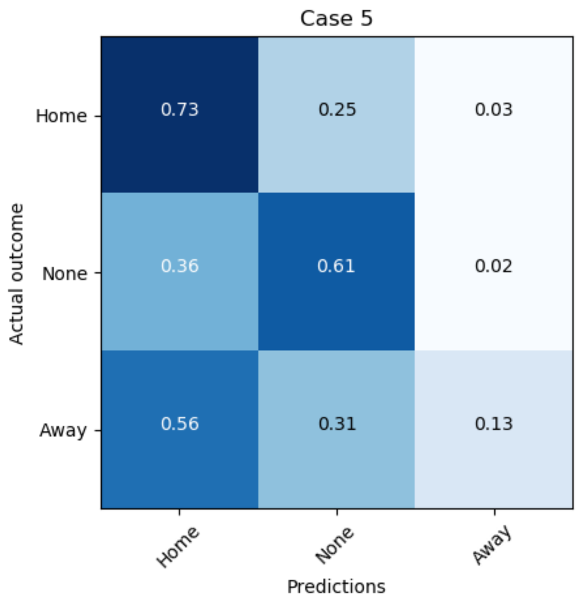


Figure 5.5: Case 5 - Confusion matrix, the normalized row sum is equal to 1.

Case 2 predicted rather well when the home team actually scored the next goal, and when there were no more goals in the game. It struggled more with producing the correct prediction when the away team was the next goal-scoring team, where 49% of the predictions where incorrectly on the home team. When the away team was the actual outcome, Case 2 predicted correctly only 32% of the times. Case 5 struggled even more with predicting correctly when the away team was the correct outcome, predicting this only 13% of the time. However, it performs considerably better than Case 2 when it comes to the two other outcomes. When the home team scoring the next goal is the actual outcomes, Case 5 predicts correct 73% of the times. Even though Case 5 almost exclusively predicts wrong when the away team is the next goal-scoring team, the overall accuracy is better than with Case 2. The overall accuracy with the test set for Case 2 and Case 5 was 0.50 and 0.53, respectively.

5.3 The 90th-Minute Betting

The yields from the different simulations had varying results, but some were exceptionally high. The highest yield achieved across all cases and simulations were when betting only at the 90th minute using the Kelly strategy, in Case 1. This achieved an average yield of 54.11, where the worst and best runs had yields of 34.29 and 72.95, respectively. Looking more detailed into this betting simulation reveals that this might not be the best strategy.

The highest yield was achieved in a run that only placed three bets during a total of 353 games. All the bets were placed on that none of the teams would score a goal in the remaining time, and at the 90th minute, the probability for this is rather high. The worst run, with a yield of 34.29, placed a total of 20 bets. 19 of these bets were placed on none, while the remaining bet was placed on the home team. The bet on the home team was the only bet that lost and the only bet that was placed on either of the teams scoring in all the five runs.

Using this strategy means the bettor has to be patient. This requires the bettor to wait for the bets to be feasible, which previously mentioned does not happen very often during the 353 games. If the bettor is patient and able to follow the odds at every game to check if a bet is feasible, making a profit is possible. However, for a bettor, placing only three bets during 353 games is probably not very fulfilling for the bettor's satisfaction. On the other hand, creating an automated system for this is possible, but placing only three bets is not sufficient to get a statistically significant result.

5.4 Feature Impact

This section will look at how the predictions in a match vary with the different cases, as different features are used. Adding EGRT and xG to the data set had a big impact on the predictions. A big difference between the EGRT values would make the model choose the team with the greatest value to be the next goal scorer. With a small difference between the EGRT values, the xG values would be the deciding factor. Match events, on the other hand, did not have a major impact on the predictions, compared to EGRT and xG. Match time was also used as a feature in every case, and proved to be useful to discover the increase of the probability of no more goals throughout a match. Using the match time as a feature was redundant in the cases where EGRT and xG were included. The EGRT was already using the time to calculate its values, in other words, time is already represented in the EGRT value.

Figures 5.6 to 5.8 show the probability distribution during a match, with Case 1 to 3. These cases use different features, which are impacting the predictions. The match is between AFC Bournemouth and Wolverhampton Wanderers, where AFC Bournemouth played at home. The

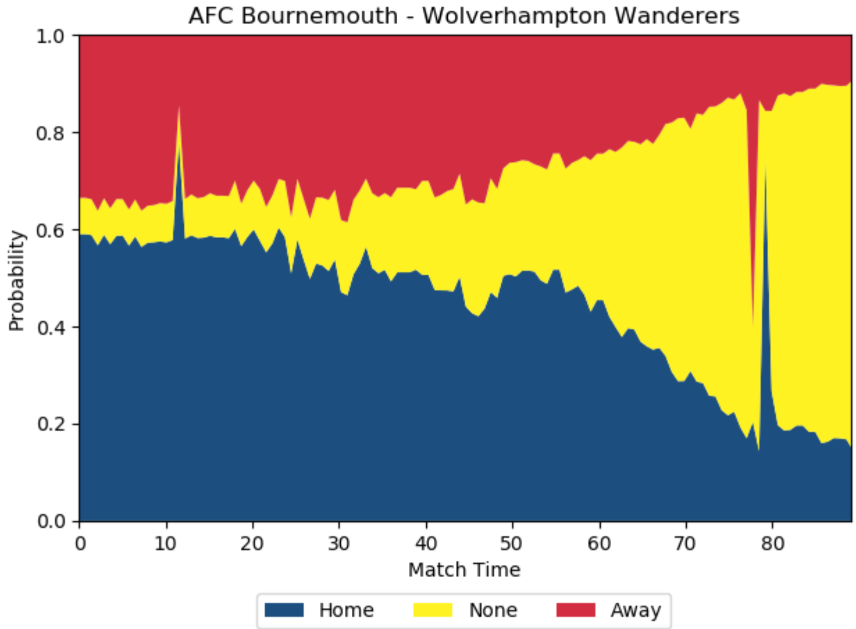


Figure 5.6: Probability distribution during a match for Case 1. Case 1 uses the EGRT value, the xG value and match events as input.

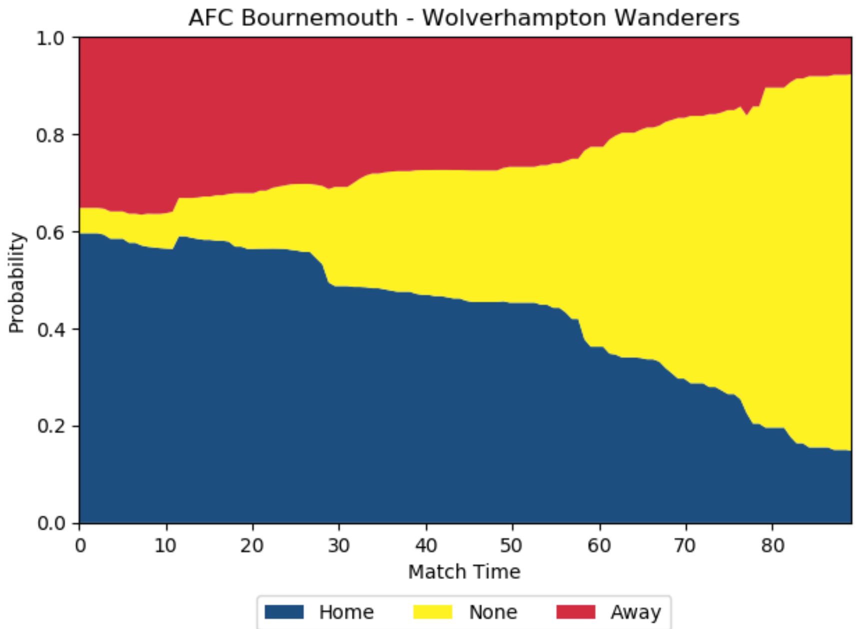


Figure 5.7: Probability distribution during a match for Case 2. Case 2 uses the EGRT and xG values as input.

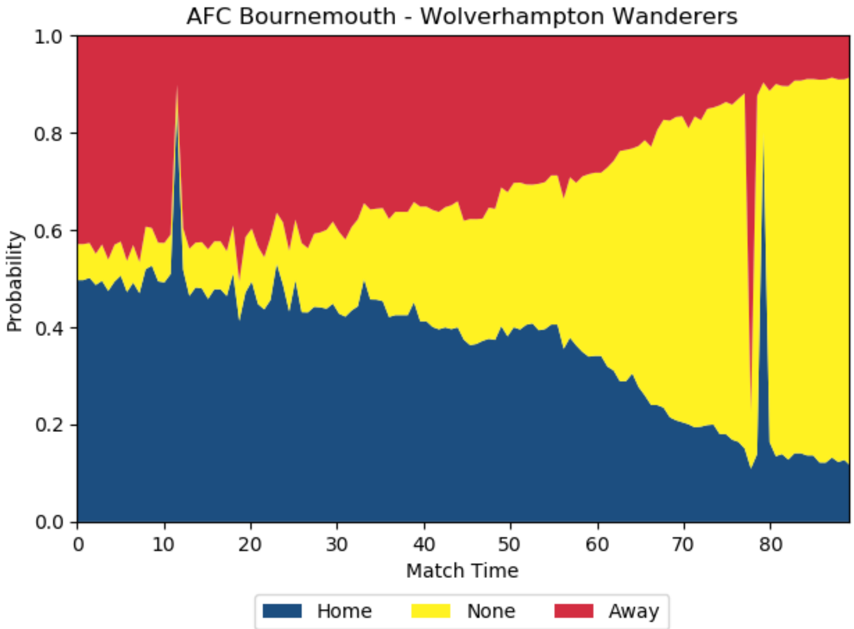


Figure 5.8: Probability distribution during a match for Case 3. Case 3 uses only match events as input.

match ended with both teams scoring one goal each, both on a penalty. In all three cases, the home team is being favored as the team that scores the next goal. Case 3 gives the home team a probability of being the first goal scorer around 0.5, at the beginning of the match. This model does not use the EGRT and xG values, but only match events as input. In Case 1 and 2, which use the EGRT and xG values as input, the initial probability is around 0.6. This is caused by AFC Bournemouth having a larger EGRT value. The EGRT is almost even after 20 minutes, but the home team's xG value is much larger than the away team's, indicating that they have had more chances up to this point. Therefore, AFC Bournemouth is still considered being the most likely team to score the next goal.

Right around the 15th-minute mark, we can see a large spike for the home team and a decrease for the enemy team, caused by a penalty shot for the home team. This can only be seen in the figures for Case 1 and 3, as Case 2 does not use match events as input. Even though Case 2 does not use these, increases in probability are still visible for the team that was awarded the penalty. However, this increase occurs right after the penalty, which is caused by a significant increase in the team's xG, as a penalty is considered a big chance. Two other spikes occur at the 80th-minute mark, both caused by penalty shots for away team and home team. The penalty shots causes the probability to increase to just below 0.8, which corresponds well to the average percentage of successful penalty shots at 76.8% [Hawerchuk, 2010]. With the first penalty, the probability for the awarded team to score the next goal increases at the expense of the opposing team. For the two other penalties, occurring in the later stages of the match, the probability now increases at the expense of the probability for no more goals. Penalties are the only event that changes the predictions this substantially.

From the figures for Case 1 and 3, we can see there are no other major spikes caused by events besides the ones caused by the penalties, only a lot of minor ones. These are created by match events, such as cards and corners. However, these smaller spikes in the predicted probabilities can be enough to decide if a bet is to be placed or not. Further, we can see from all three graphs that the probability for no more goals in the game increases as the match time increases. This naturally applies to most of the matches, as there is less time for the teams to score a goal. For Case 2, the team's probability decreases smoothly without any spikes, as the match time increases and the EGRT values gradually decrease. However, as mentioned, Case 2 has a couple of bumps in its smoothly-decreasing graphs for the teams' probability. This can be caused by a penalty, or other types of chances created that affects the xG values. This can be seen in Figure 5.7 around the 30th minute, where a decrease in the home team's probability occurs after an increase in the away team's xG value. Match events that causes spikes in Case 1 and 3 does not affect the the probabilities in Case 2, unless they increase the xG values.

5.5 Using Other Network Parameters

Case 1 is the case that uses the most data. It consists of data from all the leagues and uses all the features as input. However, in Betting simulation 1, Case 1 was one of the worst cases in terms of the yield. The seemingly best network setup, the one achieving the highest validation accuracy, was with an RNN. Throughout the results in Section 4.3, we have seen that the highest validation accuracy does not necessarily generate the highest yield. Running the betting simulations on Case 1 with another setup, that initially had a lower validation accuracy, might not give poorer results.

The best setup using a feedforward network for Case 1, was the fourth best overall. Choosing the combination of network parameters for this setup gives better results than when the parameters achieving the highest accuracy were used. The results from Betting simulation 1 are shown in Table 5.1. The results are remarkably better than previously achieved with Case 1. All the strategies achieve a positive average yield, which was previously only achieved with two cases. In addition, the fixed return strategy and the variance-adjusted strategy achieved a positive yield on their worst run. This has previously happened in two cases as well, but not while the average yield was positive with all strategies.

Table 5.1: Yields, in percent, from different betting strategies with single bets placed at random time points in each match.

	Fixed bet	Fixed return	Kelly	Variance-adjusted
Minimum	-2.91	2.18	-4.03	6.29
Maximum	11.05	11.73	8.29	13.17
Average	4.33	6.99	1.95	9.33

The results from the Betting simulation 2, also differs from the previous results with Case 1. This is shown in Table 5.2. Positive yields are now generated when betting in the 15th and 30th minute, which previously produced negative yields. The yield is still positive when betting at the end of the game, as before, but not with all strategies. Using the Kelly strategy at the 90th minute generates a yield of -21.71. This was previously as high as 54.11. As the results from Section 4.3 indicates, the Kelly strategy can give varying results.

Table 5.2: Yields, in percent, when placing bets at fixed time points in each match.

Min	Fixed bet	Fixed return	Kelly	Variance-adjusted
15	3.68	2.99	2.21	2.61
30	7.89	9.21	0.37	10.54
45	-12.50	-9.34	-12.40	-7.46
60	-0.64	-0.79	-9.72	-1.88
75	0.34	2.27	-6.53	2.89
90	-8.76	2.57	-21.71	5.19

The results from the last simulation, Betting simulation 3, are still pretty poor. Table 5.3 displays this. Both the average yields and the maximum yields are worse with all strategies. The only improvement on the average yields is with the variance-adjusted strategy, that goes all in on new predictions, which increased from 0.19 to 0.35. In contrary to Betting simulation 1 and 2, the results from Betting simulation 3 are not better than what was previously achieved.

Table 5.3: Yields, in percent, when possibly placing multiple bets in each match.

	Continuous Kelly 1	Continuous variance-adjusted 1	Continuous Kelly 2	Continuous variance-adjusted 2
Minimum	-8.98	-1.26	-7.75	-1.05
Maximum	-3.73	1.32	-2.89	1.30
Average	-6.11	0.32	-4.92	0.35

5.6 Discussion

Looking at the results in Section 4.3, we can see that the results from the different cases are varying. In the first betting simulations, where one bet was placed at a random time point in each match, the variance-adjusted strategy generated the highest yields in general. Two of the cases, Case 3 and Case 6, also had positive yields on their worst run, while the average in these cases was also better than the baselines. Only the Kelly strategy achieved a higher average yield in one of the cases, but this strategy had a negative yield with most of the cases.

Further, the model with the highest validation accuracy does not necessarily give the best yield. The variance-adjusted strategy, which was the best strategy overall, showed that Case 3 is the most successful case when it comes to average yield. As mentioned, this case also had a positive yield on its worst run. However, this is the case with the worst validation accuracy by far, out of all the cases. Case 6 is considered the best case when it comes to yield, compared to the two other cases using only matches from the Premier League. As with Case 3, this case has the worst validation accuracy out of the three cases using only matches from the Premier League. Common for Case 3 and Case 6 is that both cases only use match events as input, which we saw in Section 5.4 resulted in a lot of spikes for the predicted probabilities. These small spikes, caused by different match events, could be enough for a betting strategy to decide if a bet is to be placed or not.

The indication that the highest validation accuracy does not give the highest yield, is also demonstrated in Section 5.5. This covers our test with Case 1 and new network parameters, which had a lower validation accuracy than the initial network setup. This revealed that the yield from the network with these parameters was superior to the previous network setup. A solution for this could be running a betting simulation on the validation set, and choosing the set of network parameters that generated the highest yield. However, we did not have odds available for the validation set, and could therefore not run any betting simulations based on this.

Looking at Betting simulation 2 with the variance-adjusted strategy, gave varying results. In general, a positive yield was achieved when betting late in the match with the three first cases, which consisted of data from all five leagues. For the three last cases, which consisted of data from only the Premier League, a positive yield was achieved when betting early in the match. Case 3 is the exception, which generated a positive yield at all time points, except when betting at the 60th minute. However, only betting at one time-point may not be the most realistic betting scenario. As we have seen in Section 5.3, betting only at the 90th minute greatly restricts the number of bets placed. On average, under 15 bets were placed in each of these simulation, out of 353 matches.

The continuous betting gave rather poor results, compared to Betting simulation 1. Only three of the cases had a positive average yield using a variant of the variance-adjusted strategy, while the first betting simulation had a positive yield in all cases. This indicates that the proposed strategies for continuous betting might not be the optimal strategy.

Chapter 6

Conclusion and Future Work

This chapter concludes the thesis, and proposes some suggestions for future work.

6.1 Conclusion

Throughout this project, the tests done have been in an attempt to find answers to the research questions from Section 1.2, and to try to reach the main goal of the project.

Research question 1 *What features are important when using machine learning to predict the next goal-scoring team?*

The accuracies from the different cases revealed that the cases using only matches from the Premier League achieved the highest accuracies. The highest accuracy, by far, was achieved by Case 5. This case used only matches from the Premier League and the EGRT and xG values as input. For the three cases including matches from all the leagues, Case 2 achieved the highest accuracy. This case used the EGRT and xG values as input, as with Case 5. Based on this, it seems like only including a single league gives the highest accuracy. Further, using only the EGRT and xG values as input gives higher accuracy than the cases where the match events are included. In addition, the accuracies achieved by all the cases were higher than the baselines.

Research question 2 *How can the predictions be used to gain profit when betting?*

When placing only single bets in each match in Betting simulation 1, positive yields were often achieved. Overall, the variance-adjusted strategy gave the best results. In the first betting simulation, when bets were placed at a random time point, this strategy achieved a positive yield in all cases. Contrary to the cases achieving the highest accuracies, the cases using match events and excluding the EGRT and xG values as input gave the highest yields. This was Case 3 and Case 6. These two cases generated positive yields even on their worst run, and their average yield is also higher than the baselines.

The continuous betting did not achieve positive yields as frequent as Betting simulation 1. Case 4 and 6 had positive average yields with all strategies, but only one of the strategies in Case 6 had a positive yield in its worst run. Case 4 achieved average yields that were higher than the baselines using the Kelly strategies. Case 4 and 6 included only matches from the Premier League. The poorer yields in continuous betting can indicate that the strategies might not be optimal.

Goal *Use machine learning to predict next goal-scoring team in a football match, in order to get an edge on bookmakers,*

Even though there is a lot of uncertainty in football, and the bookmaker's odds are well calibrated, we have shown that it is possible to beat the bookmakers based on predictions made using machine learning. The variance-adjusted strategy, which gave promising results, uses the predicted probability from the neural network to decide how much to place in a potential bet, which was the overall goal with the project.

6.2 Future Work

There have been some limitations to the project, which can be explored in future work.

- **Parameters-selection criteria:** We have seen that the combination of network parameters that achieves the highest validation accuracy, not necessarily gives the highest yield on a test set. A possibility would be to run a betting simulation on the validation set and choose the network parameters that achieve the highest yield. We did not have the odds available for the matches in the validation set and were therefore not able to test this.
- **Features:** Additional features should also be experimented with. Some of these features that could be considered are crosses, ball possession and duels won. We did not use these features as we only included match events that had a timestamp, which these did not have in the data provided by Sportradar. Other features such as kilometer covered by the players, heat maps of where the players have been, player ratings and lineups could also be considered if the data is available.
- **More data:** Having more matches in the test set would also give results that are more statistically significant. We only had odds from the matches played in January and February in the current season available for the simulations.
- **Continuous betting:** The results from Betting simulation 3 gave rather poor results. Given the high accuracies achieved, and the good results from Betting simulation 1, these results might indicate that the proposed strategies for continuous betting might not be optimal. For future work in this field, finding better strategies for the continuous betting can be a good starting point.

Bibliography

Ammon, C. (2016). How I use "Poisson Distribution" to successfully predict winning soccer outcomes. Available at http://fspinvest.co.za/articles/strategies/how-i-use-poisson-distribution-to-successfully-predict-winning-soccer-outcomes-7331.html. Accessed: 2018-09-12.

Australia Sports Betting (2010). Kelly Criterion - Part 3d - Fractional Kelly Betting. Available at http://www.aussportsbetting.com/2010/07/27/kelly-criterion-fractional-kelly-betting/. Accessed: 2019-03-26.

Betacademy (2019). ROI and Yield in Betting - How to Calculate Them. Available at https://www.betacademy.com/roi-and-yield-in-betting-how-to-calculate-them/. Accessed: 2019-05-23.

Boice, J. (2018). How Our 2018 World Cup Predictions Work. Available at https://fivethirtyeight.com/features/how-our-2018-world-cup-predictions-work/. Accessed: 2018-10-25.

Brownlee, J. (2017). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. Available at https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. Accessed: 2019-05-23.

Cronin, B. (2017). Poisson Distribution: Predict the score in soccer betting. Available at https://www.pinnacle.com/en/betting-articles/Soccer/how-to-calculate-poisson-distribution/MD62MLXUMKMXZ6A8. Accessed: 2018-09-13.

Dixon, M. J. and Coles, S. G. (1997). Modelling association football scores and inefficiencies in the football betting market. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 46(2):265–280.

Donges, N. (2018). Recurrent Neural Networks and LSTM. Available at https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5. Accessed: 2019-05-22.

FiveThirtyEight (2018). 2018 World Cup Predictions. Available at https://projects.fivethirtyeight.com/2018-world-cup-predictions/?ex_cid=endlink. Accessed: 2018-11-06.

Haugård, M. E. S. and Vu, K. L. (2018). Computational models for live-prediction of football matches. Available at https://www.dropbox.com/s/k0es29ecdb8v9pe/Prosjektoppgave.pdf?dl=0. Accessed: 2019-05-06.

Hawerchuk (2010). Soccer: Penalty Kick Percentage across Leagues. Available at https://www.arcticicehockey.com/2010/7/9/1560962/soccer-penalty-kick-percentage. Accessed: 2019-05-20.

Kelly, J. L. (1956). A new interpretation of information rate. *IRE Transactions on Information Theory 2(3)*, pages 185–189.

Langseth, H. (2013). Beating the bookie: A look at statistical models for prediction of football matches. In *SCAI*, pages 165–174.

Maher, M. J. (1982). Modelling association football scores. *Statistica Neerlandica*, 36(3):109–118.

Moroney, M. J. (1951). *Facts from figures.* Penguin books, 3 edition. Revised 1962.

Pettersson, D. and Nyquist, R. (2017). Football Match Prediction Using Deep Learning. Available at http://publications.lib.chalmers.se/records/fulltext/250411/250411.pdf. Accessed: 2018-10-15.

Pinnacle (2019). How to calculate live odds for soccer betting. Available at https://www.pinnacle.com/en/betting-articles/Soccer/how-to-calculate-live-odds-for-soccer-betting/2Y42JJBF7P9GB2P2. Accessed: 2019-03-08.

Premier League (2019). Spurs v Arsenal, 2018/19. Available at https://www.premierleague.com/match/38594. Accessed: 2019-03-11.

Reep, C., Pollard, R., and Benjamin, B. (1971). Skill and chance in ball games. *Journal of the Royal Statistical Society. Series A (General)*, pages 623–629.

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. and Norvig, P. (2016). *Artificial Intelligence: A Modern Approach.* Berkeley, 3 edition.

Shibuya, N. (2018). Demystifying Cross-Entropy. Available at https://towardsdatascience.com/demystifying-cross-entropy-e80e3ad54a8?fbclid=IwAR0EbEymjvXyPywfHcMWuUc0Wb7PbvDhelob0Ir_A5URDbN40qDIsfiDrP0. Accessed: 2019-05-23.

Silver, N. (2019). When We Say 70 Percent, It Really Means 70 Percent. Available at https://fivethirtyeight.com/features/when-we-say-70-percent-it-really-means-70-percent/. Accessed: 2019-05-15.

Skalski, P. (2018). Preventing Deep Neural Network from Overfitting. Available at https://towardsdatascience.com/preventing-deep-neural-network-from-overfitting-953458db800a. Accessed: 2018-12-01.

Smith, A. (2017). Sky Sports bust common football myths: Home advantage? Available at https://www.skysports.com/football/news/11096/10955089/sky-sports-bust-common-football-myths-home-advantage. Accessed: 2018-11-08.

Sportradar (2019). Soccer Extendend v3. Available at https://developer.sportradar.com/docs/read/football_soccer/Soccer_Extended_v3. Accessed: 2019-03-07.

Understat (2019a). Celta Vigo 0 - 1 Real Betis. Available at `https://understat.com/match/10208`. Accessed: 2019-03-12.

Understat (2019b). xG stats for teams and players from the TOP European leagues. Available at `https://understat.com/`. Accessed: 2019-03-12.

van Lier, J. (2018). The Most Popular Sports to Bet on Today. Available at `https://bitcoinchaser.com/bitcoin-sportsbook/most-popular-sports-to-bet-on-today`. Accessed: 2018-10-12.

Venkatachalam, M. (2019). Recurrent Neural Networks. Available at `https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce`. Accessed: 2019-05-21.

NTNU
Norwegian University of
Science and Technology