Master's degree thesis

LOG950 Logistics

Testing the betting market efficiency with the use of heuristics.

Hans Jacob Brun

Number of pages including this page: 68

Molde, 23.05.2019

Molde University College
Specialized University in Logistics

Mandatory statement

Each student is responsible for complying with rules and regulations that relate to examinations and to academic work in general. The purpose of the mandatory statement is to make students aware of their responsibility and the consequences of cheating. Failure to complete the statement does not excuse students from their responsibility.

*Please complete the mandatory statement by placing a mark **in each box** for statements 1-6 below.*		
1.	**I/we hereby declare that my/our paper/assignment is my/our own work, and that I/we have not used other sources or received other help than mentioned in the paper/assignment.**	☒
2.	**I/we hereby declare that this paper** 1. Has not been used in any other exam at another department/university/university college 2. Is not referring to the work of others without acknowledgement 3. Is not referring to my/our previous work without acknowledgement 4. Has acknowledged all sources of literature in the text and in the list of references 5. Is not a copy, duplicate or transcript of other work	Mark each box: 1. ☒ 2. ☒ 3. ☒ 4. ☒ 5. ☒
3.	**I am/we are aware that any breach of the above will be considered as cheating, and may result in annulment of the examination and exclusion from all universities and university colleges in Norway for up to one year, according to the Act relating to Norwegian Universities and University Colleges, section 4-7 and 4-8 and Examination regulations** section 14 and 15.	☒
4.	**I am/we are aware that all papers/assignments may be checked for plagiarism by a software assisted plagiarism check**	☒
5.	**I am/we are aware that Molde University College will handle all cases of suspected cheating according to prevailing guidelines.**	☒
6.	**I/we are aware of the University College's rules and regulation for using sources**	☒

Personal protection

Personal Data Act

Research projects that processes personal data according to Personal Data Act, should be notified to Data Protection Services (NSD) for consideration.

Have the research project been considered by NSD? ☐**yes** ☒**no**

- If yes:

Reference number:

- If no:

I/we hereby declare that the thesis does not contain personal data according to Personal Data Act.: ☒

Act on Medical and Health Research

If the research project is effected by the regulations decided in Act on Medical and Health Research (the Health Research Act), it must be approved in advance by the Regional Committee for Medical and Health Research Ethic (REK) in your region.

Has the research project been considered by REK? ☐**yes** ☒**no**

- If yes:

Reference number:

Publication agreement

ECTS credits: 30

Supervisor: Lars Magnus Hvattum

Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).
All theses fulfilling the requirements will be registered and published in Brage HiM, with the approval of the author(s).
Theses with a confidentiality agreement will not be published.

I/we hereby give Molde University College the right to, free of charge, make the thesis available for electronic publication: ☒yes ☐no

Is there an agreement of confidentiality? ☐yes ☒no
(A supplementary confidentiality agreement must be filled in)
- If yes:
Can the thesis be online published when the period of confidentiality is expired? ☐yes ☐no

Date: 23.05.2019

Abstract

A way of investigating inefficiency in the betting market is to develop a prediction model and check whether it outperforms the market via simulated betting, for instance by the use of ordinal regression models. A compass search heuristic has been created to fine tune the parameter values obtained by the regression model “Ordered Logistic Regression”. This to check whether it is possible to detect inefficiencies in the betting market and also if there exist any categorical wrong settings of parameters obtained by the statistical model. Even though the compass search was able to outperform OLR in accuracy, yielding a higher average return, no categorical wrong settings in OLR were found.

KEYWORDS: FOOTBALL, BETTING, HEURISTICS, OLR, MARKET EFFICIENCY, DIRECT SEARCH

Contents

1.0 Introduction

Optimizations techniques are extensively used in many fields. In operations research (OR) one usually thinks of optimization as a quantitative approach to solve problems related to production, inventory or vehicle routing. In this thesis a central technique in OR, namely heuristics, will be applied in the effort to improve statistical model used when predicting the outcome of football games. (For the potential American reader(s), it should be mentioned that football here is understood as soccer). The usage of statistical methods in sport outcome predictions are quite common and most bookmakers rely on this amongst other things when setting the odds for different outcomes.

Due to technological improvements everyone is now just a few clicks away from betting on outcomes of everything from prime minister elections to football matches. The United Kingdom Gambling Commission reports an online gambling revenue of £5.319 billion annually (Gambling Commission, 2018) where a considerable share is betting. In addition, does non-remote betting also contributes a revenue of £3.254 billion annually. Because of the wide variety of opportunities of what and where one can place bets, and the combination of amateurs and professional actors in the market one could imagine that the prices, here understood as the odds, does not correctly reflect all the relevant public information available. The questions whether this is true or not are related to market efficiency, which can be read about in Chapter 3. And if an assumption of inefficiency in the odds market is true, is it possible to develop a trading strategy utilizing the fact that the odds are not set correctly? That is what to be investigated in this paper. Prediction models are of interest in OR because many decisions within production and inventory management are based on forecasting, for instance demand, or how prices on raw materials changes. At first glance could it look like this thesis do not relate to either of this, but the techniques applied are universal, in the sense that one uses heuristics to improve an existing model. The transferability to other data within areas beside from sports is unquestionable.

A way of investigating inefficiency in the betting market is to develop a prediction model and check whether it outperforms the market. Typically calculating the probability for the different outcomes of a football match based on information which might affect the game, do this for an entire dataset of matches and compare estimated probabilities with probabilities given by the bookmaker. Through simulation it is possible to conclude that

this is not enough to outperform the market. But by applying a heuristic on top of the prediction model, it could be possible to find indications that the betting market is insufficient in some way.

The paper is structured into three main parts. Where the first part contains Chapter 2. Research Question, Chapter 3. Theory, which is, together with Chapter 4. The Heuristic Approach to Optimization, a theoretical part to familiarize the reader with the concepts necessary to understand what is being done in the thesis. The second part contains Chapter 5. Experimental Setup, which includes the choice of data and other technicalities necessary in the reach for an answer for the research questions. The third part contains Chapter 6. The Search, where the tests are done, and results are discussed. Chapter 7. Concluding Remarks where the conclusions are drawn. And the final chapter, Chapter 8. Acknowledgements. The thesis will provide pseudo codes for simulations and heuristics, in addition to code attached in the appendix.

2.0 Research Question

This thesis will investigate the betting market and its efficiency. There are two types of tests that are usually performed based on the whether the market is weak or semi-strong in its efficiency. The thesis will look into whether the market is consistent with the hypothesis of semi-strong information efficiency, which for the betting market, or any market for that matter, can be defined the following way:

«Semi-strong efficiency means that the return on a bet based on public information must be the same, in terms of cost/risk, as that on a bet that has not been based on public information» (Bernardo, Ruberti et al. 2018)

When dealing with such issues as investigating the relationship between two, or in this case, many variables, is it common to use statistical methods and moreover regression models. Further on, when testing the efficiency in the betting market, and in this particular case, the football betting market, a model known as ordered logit regression, from now on OLR, has proven to yield good results, but maybe not perfect.

The research questions the thesis aims to answer is the following:

1: Are there certain kinds of inefficiencies that may be detected using a heuristic approach that would not be detected when relying on maximum likelihood estimation, which the OLR is based on?

2: And further on, if the heuristic method does not find categorical "wrong setting" of odds, is it possible to manipulate the data to show that the model is able to detect these wrong settings if they existed?

3.0 Theory

3.1 Market Efficiency

The efficient market hypothesis was developed parallel by Eugene Francis Fama and Paul A. Samuleson around 1965. Fama says the following about his hypothesis:

In an efficient market, competition among the many intelligent participants leads to a situation where, at any point in time, actual prices of individual securities already reflect the effects of information based both on events that have already occurred and on events whitch, as of now, the market expects to take place in the future.
-Speech by Eugene Fama (1965).

The implication of this statement is that it is impossible to make money risk free, due to the markets ability to incorporate historical and expected future events into the asset's prices. The theory can be divided into three variants. Weak, semi-strong, and strong. A weak form of efficient can be understood as if current prices reflect the information stored in historical prices (Poshakwale ,1996). Semi-strong takes additionally current public information into account and in a strong efficiency market the information which is omitted for the public is also incorporated. Which means that even inside information is useless to take advantage of because the price already reflects that "cut-off from the public" information.

The area this thesis will take a more thorough look at is the efficiency in the betting market concretized by football matches and their odds. Where the odds of a given outcome of a match can be reviewed as its price. A statistical model known as ordered logit regression (OLR) will be applied to create a statistical distribution and a heuristic will be used to

improve the parameters obtained by OLR, hoping to outperform the market via simulated betting.

With the use of public information when estimating the probabilities for the different outcome, as done in the thesis, the hypothesis of semi-strong market efficiency suggests that it should not improve the precision of the predictions (Gross, Rebeggiani, 2018) and therefore the simulation should not yield a return on investment significantly different from the expected returns on blind betting.

3.2 The Betting Market

The verb *bet* is defined by the Cambridge Dictionary the following way: "to risk money on the result of an event or a competition, such as a horse race, in
the hope of winning more money". The betting market as a market in more fixed terms has at least been around since ancient Rome (Gross, Rebeggiani 2018). But even if betting markets existed in unregulated ways since very early are the opportunities we today take as for granted, a result of a continues war of interest between gambling interest and reformers (Sauer, 1998). Andreff and Szymanski suggests the betting opportunities we now experience is a result of secularization (Handbook of the Economics of Sports, 2006).

In the same way that the stock market needs stock exchange as a platform for buying and selling stocks, the betting market needs the same type of platform. The bookmaker will typically give odds on several outcomes of certain events and the participants can choose to place a bet roughly based on either two terms: They believe that the odds is set higher than the underlying probability, which means that the expected return is positive. Or they base their bets on loyalty to a team, a player. In other terms; just for fun.

There is, however, one big difference between the stock market and the betting market. When one trades stocks, the buyer bets against the seller. The stock exchange serves as a neutral 3rd part, earning their money on a brokerage from the trade. When a gambler places a bet, on the other hand, the bet is between the bookmaker and the gambler. So, the bookmaker sets the prices and not solely the market itself as it occurs in a stock trade. There exists some market dynamics in odds setting as well, which will be discussed in the next section, but in general one can say that a bookmaker enables the gamblers to place bets and they also stands on the other side of the bet. This is important for two reasons. Since each bookmaker can be viewed as its own market, i.e. the odds for a certain event

given at one bookmaker does not have to be the same as at a different bookmaker, and gamblers stand free to investigate which bookmaker provides the best odds at a given time, the bookmakers will strive for the best possible odds to attract customers. On the other hand, since the bookmakers are the ones who is left with the bill if a bet goes against their interest, are they not interested in giving better odds than they estimate the underlying probability to be for the event.

3.3 Odds and Bookmakers

Odds are defined as the inverse of the probability and can be expressed the following way:

$$O(x) = \frac{1}{p(x)} \qquad (1)$$

Where $p(x) =$ the probabilty of event x occouring. The odds serve two functions, it gives an impression of how likely an incident is to occur, and the odds are also used as the factor when calculating the return if a bet goes in the favor of the actor who placed the bet. Let us say a gambler places a bet of 100 units on a specific outcome of an event with the odds of 5. This tells us the two things mentioned above. The odds setter considers the probability of this event to occur to be $1/5 = 20\%$. And that if the gambler wins the return will be $5 * 100 = 500$.

Since this paper deals with odds in football matches, the focus will now move over in that direction. In a football match there is only have three outcomes, and it is a 100% probability that one of them will occur. Therefor the sum of the probabilities must be 1. This is called the true odds. But in a real betting situation a bookmaker will rarely provide true odds, because the bookmaker needs a margin of protection, known as the overround. Meaning that the sum of the probabilities implied by the odds for the three different outcomes in a football match adds up to more than 1. The bookmakers will arrange the odds in a way that they will pay out the same no matter the odds and collect the overround (Haigh 1999). An arbitrary game, Fulham-Everton played 13.04.2019 had the odds 4.70, 4.15 and 1.76 for (the outcomes) home, draw and away, respectively. When translated into probabilities and added together $\frac{1}{4.7} + \frac{1}{4.15} + \frac{1}{1.76} = 1.0219$, meaning that if 21.28 units is played on home win, 24.01 on a draw, and 56.82 units on away win, the bookie has to pay out exactly 100 units, no matter what the outcome will be. And is left with 2.19 units in profit.

Odds will also change as the demand in placing bets of a certain result increases. One could maybe review this as standard market dynamics, where the seller understands that the goods being sold is too cheap and the price increases until demand stabilize. When talking about odds, it is important to stress the fact that it work opposite of normal pricing. Where the odds will decrease as the demand increases. We know that high odds imply two things, low probability and high return. If enough people still want to place a bet on the outcome with high odds/low probability, signalizes this that actors in the market believes the probability of the outcome is higher than implied by the odds and is therefore worth taking the risk given the return. Since the bookmakers are the ones who has to pay out if the bet goes in favor of the gambler will they lower the odds to guard themselves.

In summary, there are three factors that determine how bookmakers set odds. First and maybe most important a statistical method, possibly similar to the one presented in this paper. Second are the football experts. One study suggests that the consensus of subjective odds setting being less accurate than statistical models, is wrong (Forrest, Goddard, Simmons, 2005). And the third is the market forces discussed in the previous section.

3.4 Elo Rating

The Elo rating system is originally used to calculate the relative strength for a chess player and was created by professor Arpad Elo in 1961. (Elo, A. E. 1961). Glickman and Albyn C. Jones (1999) says the following about Elo and how it is calculated in their paper *"Rating the Chess Rating System":*

"The fundamental assumption of Elo's rating system is that each player possesses a current playing strength, which is unknown, and that this strength is estimated by rating. In a game played between players with (unknown) strengths R_A and R_B, the expected score of the game for player A is assumed to be

$$E_A = \frac{1}{1 + 10^{-(R_A - R_B)/400}} \qquad (2)$$

"

The expected score for player B will then be $1 - E_A$. The function does not take the probability of a draw into account because draws are treated as a half win and a half loss

for each player. When these probabilities are calculated and the games has settled, it is possible to calculate and update the players rating using the following formula

$$r_{post} = r_{pre} + K\,(S - E) \qquad (3)$$

Where r_{pre} is the players ranking previous to the game, S is the score of the game (1, 0.5 or 0) and K is a factor telling how much the impact the game should have. The value of K is divided into three categories depending on your current rating. If the existing Elo-rating for a player is greater than 2400 a $K = 16$ value is used, for players with the rating interval of 2100-2400, $K = 24$ and for those players rated sub 2100, $K = 32$.

A football match is similar to a chess game in the sense that there are three outcomes, and that the teams/players will have different strength. Therefore, it would be convenient to have a similar type of rating system for football teams. Some aspects, however, differ between the two sports. For instance: in chess a win is a win, it is not taken into consideration how many pieces are captured by the players or how much time is left on the clock. Just imagine if a chess game won by a player should be less valuable because of a queen sacrifice compared to a game won with the queen still intact, or if the player did not spend the allotted time to find the critical move.

In football, on the other hand, is it more nuanced. Since goals wins games, will a team's ability to score goals tell something about their strength at a given state beyond just winning the match. Even though a game won 1-0 generates the same amount of points as a game won 4-0, is the second victory clearly more impressive and tells something about this ability. A game won with four goals should count as a greater achievement than winning with one goal and should therefor give greater manifestations in the updated rating. Some attempts have been done to capture this aspect. In the article "Using ELO ratings for match results predictions in association football" (Hvattum, Arntzen, 2010) this is dealt with by making $K = K_0(1 + \delta)^{\lambda}$, where δ is the absolute goal difference and $K_0 = 10, \lambda = 1$ as fixed parameters. This is also known as the *goal based* Elo rating. And this is the method used for calculating the Elo ratings in the dataset for this thesis.

It is important to emphasize the fact that the ratings are not perfect from the get-go and need some initial runs before they become reliable. The bootstrapping procedure is applied for the purpose. In the beginning will all the teams have the same rating. Then two seasons of data is used to update the ratings. If the rating obtained in the end of the two seasons are

similar to the rating originally assigned will the process stop. If not, will the rating obtained in the end of the second season be assigned as the new rating and the procedure is repeated (Hvattum 2017). When the current Elo ratings become reliable, after the bootstrapping, it is a very useful and logical way of reviewing the current strength between two competitive teams.

3.5 Ordered Logistic Regression

Ordered logistic regression (OLR) is a statistical technique for estimating the probabilities of several outcomes where the outcomes have a clear ranking. The reason why this "ordered" approach can be applied in football betting markets is because one can order the outcome of a football match from a home team perspective where a win is better than a draw, which is again better than a loss.

One has a categorical outcome *y* which generally can take the values $1,2,\dots,K$. When football is the topic a game can have three outcomes and therefor *y* can take the values $1 =$ home win, $2 =$ draw, and $3 =$ away win. Additionally, there are *V* independent variables $x_1, x_2, \dots, x_V$ which are calculated prior to each match. All of the independent variables will be described in greater detail later in the paper, but to give some intuition can one such variable be "how many goals on average does the teams score." If a team on average scores many goals, could this increase the probability of this team taking the victory home in the next incidence also. For each variable x_i there is a corresponding parameter β_i which can be reviewed as the weight each variable should contribute when determining the probabilities for the different *y* values. The parameters to the variables are what to be estimated by the model.

The estimation also provides cutting points parameters θ_i for θ_{K-1}, which are points dividing the cumulative probability function into categories along the first axis. To fit the parameters in the OLR model, the maximum likelihood method is used for minimizing the information loss. Most statistical software packages will provide this. (Devore, Berk, 2012)

The cumulative probability distribution function for OLR

$$F(z) = \frac{1}{1 + e^{-z}} \qquad (4)$$

The conditional probabilities for the three outcomes in a football match will then be

$$\pi_1(x) = F(-\theta_1 - \beta x),\ \ \pi_2(x) = F(-\theta_2 - \beta x) - \pi_1(x),\ \pi_3(x) = 1 - \pi_1(x) - \pi_2(x) \qquad (5)$$

The distribution of the OLR is displayed below, where the cutting point $-\theta_1 - \beta x$ along the first axis will mark the probability of a home win along the second axis, and the cutting point $-\theta_2 - \beta x$ will separate the cumulative probability of a draw and an away win.

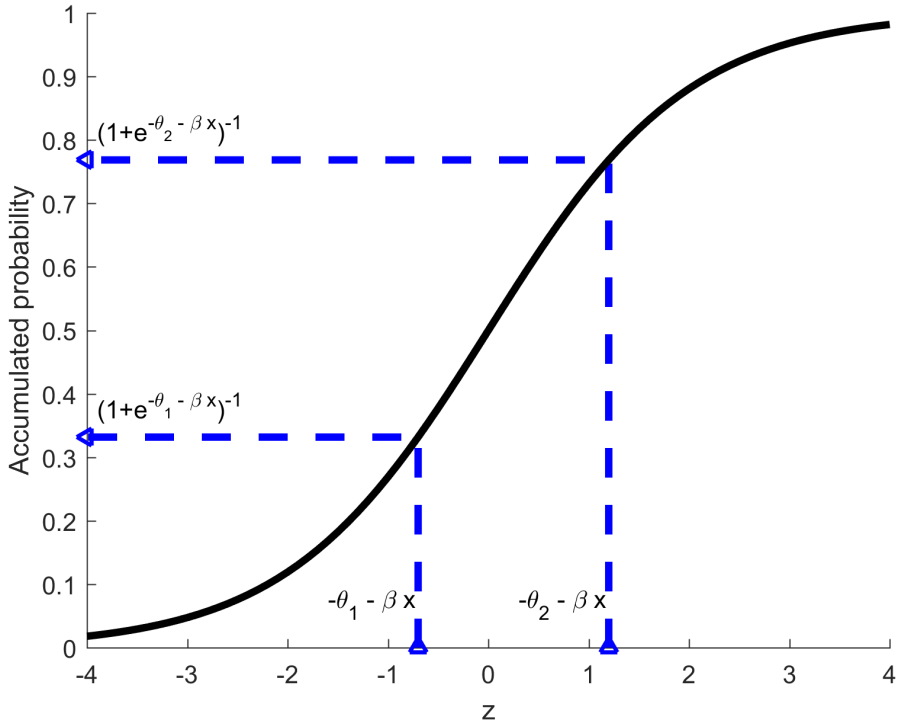

Figure 3.1

A weakness in OLR, when predicting outcomes of football matches, could be the ability to detect risk aversion of the team Hvattum (2017) i.e. the model has no ability to decrease or increase the distance between θ_1 and θ_2 only move them parallel along the first axis. Meaning that if the probability for a draw is skewed to one of the sides, the ratio between the probability of home team victory and away team victory will changes, even nothing implies this. For instance, it is possible to imagine a situation where one team need to obtain a draw or better, and the other team do not care. The relative strength between the two teams has not changed, they still have the same players and general assets, but the chances of a draw has dramatically increased. Where the team who must obtain a draw or

better will change into a defensive formation and hope for the best. These circumstances do the OLR model have a problem with detecting.

Another aspect that could be interesting to investigate is the strategy the bookmakers apply when a popular team plays. One could have the assumption that the popularity of a club and the fans belief in the team's ability to win matches is not solely based on facts. And further on assume that the bookmakers are interested in taking advantage of this. Let us use Manchester United which is according to Google-hits is about twice as big as for example Liverpool FC. If the bookmakers know that a fair number of fans is betting on Manchester United based on their passion for the club and their belief in their ability to win, a smart move for the bookmakers is to lower the odds, and thereby increasing the profit.

3.6 Standard Error

OLR estimates the parameter values which is used to predict the outcome of a match. For all of these parameters some kind of uncertainty is involved. The uncertainty can be quantified by the use of standard error, which is defined to be the estimated standard deviation of the parameter itself. When the estimated parameter is normal distributed, which is assumed given the large sample size in the data set, one says that the true parameter value lies within two standard deviations of the estimated parameter value. (Devore, Berk, 2012). The reason why this value is of interest is its ability to tell how good the estimation is. If the standard error is big do this mean that the interval where the true parameter value exists is larger. The standard error for the parameters will differ, meaning that some estimations are close to the true value and some are a bit further away. This interval where assumingly the true parameter value lies within will be investigated in the thesis.

4.0 The Heuristic Approach to Optimization

A heuristic provides solutions for problems that analysis is unable to solve. (Gigerenzer, 2006) Many of the problems one encounter in real life will not be solvable with standard optimization techniques, simply because of the size of the problem. If, for instance, the complexity of a scheduling problem increases, it quickly becomes impossible to verify that the scheduled order is by fact the optimal solution. Chess is mentioned previously in this

thesis, and chess engines are good example of the application of heuristics. A heuristic does not by definition guarantee more than feasibility for the solution. It is therefore up to the creator of the heuristic to make the algorithm smart enough to detect good solutions.

As hinted in the previous section is a heuristic an algorithm used to explore the solution space (see next section). According to a given set of rules will the algorithm search for solutions better than solutions already obtained. And if the search does not yield any better solutions are the search completed. A heuristic, however, is usually equipped with some "smartness" to it. This to avoid the search of getting trapped in a stationary point, or to allow it to start searching a different region of the solution space if the region previously visited generated similar solutions. The heuristic applied when tuning the parameters from the OLR falls under the category of a direct search. A direct search does not use gradient information and only needs the ordinal data about the function data, which can be understood as a ranking of the output the simulation gives. This is what makes the direct search methods appropriate for problems involving optimization based on simulation (Hvattum, Glover 2009).

4.1 Solution Space

A solution space is an area consisting of all the solutions that can be generated as a combination of the parameter values of the variables within a problem. The solution space consists of two parts, the feasible and the infeasible region. Since one solution is a combination of the different values the variables can take, can the solution space be reviewed as infinite. And even if the feasible region of the solution space is bounded will there exist infinite solutions, given that the problem which is to be solved is not an integer problem. Therefore, the construction of the search algorithm will decide the structure of the solution space where the feasible solutions are the ones that can be reached according to the rule of the algorithm. This region is called the neighborhood. In such cases as in this thesis, where the evaluation is done according to the return on the betting, will there not be any boundaries on which values the parameters can take. But it is suspected that most of the parameter values will be close to the estimated parameter values from the OLR, and a search in areas far from those will most likely lead to a disappointment (Hvattum, Glover 2009).

4.2 Compass Search

The compass search is a type of direct search method (Hvattum, Glover 2009) and the name of the search is related to the cardinal directions where one searches through the solution space by moving in a straight direction according to a given step length where the parameter values are either increased or decreased according to a rule. When dealing with a multi-dimensional space, a visualization is difficult. But to give an idea of how it works are the two limit values in the betting simulation (γ_l, γ_u) used, where the lower bound is located on the first-axis and the threshold on the second-axis. And the solution is a combination of both of them.

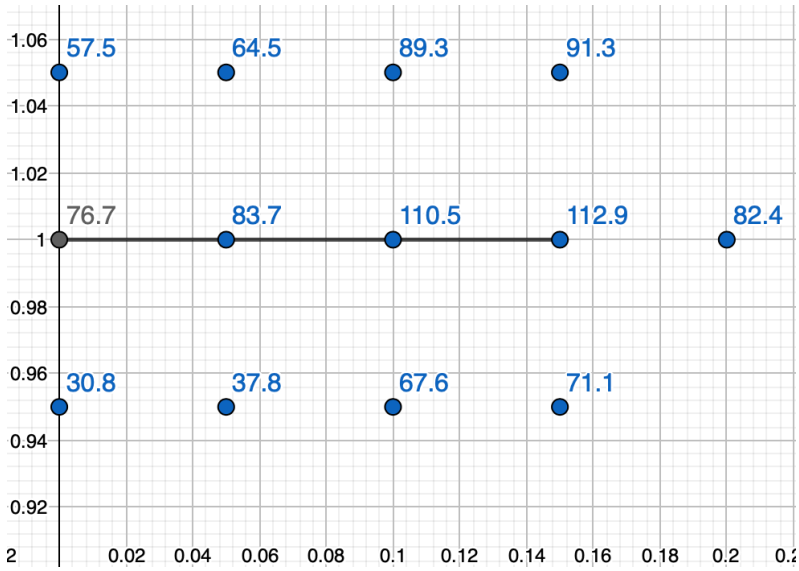

Figure 4.1

Since the probability of something never can be less than zero will the search start with three directions, increasing the threshold, decreasing the threshold or increase the lower bound. The highest value, which is understood as the highest return on the betting, occur when the lower bound is increased, and this solution is stored as best new solution. The algorithm continues in the same fashion until no further improvements can be done. And we have reached a local optimum. The rule used in this search is rather unsophisticated, by just increasing or decreasing the value of the bound and threshold by 0.05 for each iteration. But for illustrative reasons will it serve its purpose. In a real search will more sophisticated techniques usually be applied. For instance, the ability of increasing or decreasing the step length when no further improvements can be done. Also using a smarter step length. Either a percentage step length of the parameter value itself, or the standard error of the parameter.

4.3 Convergence to a Local Optimum

A heuristic which search blindfolded trough the solution space after a solution will stop at a point where no further improvements can be made according to the rule of the heuristic. We have then encountered a local optimum. An attribute of the compass search is that the solution will converge towards a local optimum. Other heuristics may allow a worsening solution in the hope of escaping a less interesting area of the solution space. This is not the case for the compass search, where each new solution is an improvement and therefore also closer to the local optima. When no further improvements can be done, according to the defined step length, can the step length be decreased, creating a new neighborhood, making it possible for the heuristic to get even closer to a true local optimum.

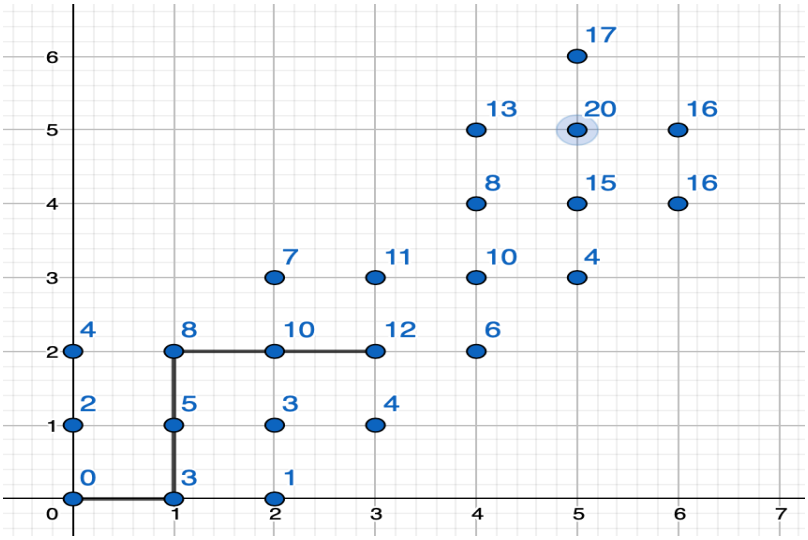

Figure 4.2

Figure *4.2* shows in total 23 solutions. Imagine that these are all the feasible solutions for a given problem, i.e. no constraints are violated. Here one sees the behavior of the compass search, starting in (0,0) and for each new step taken the value of the objective increases until the point (3,2) where no improvements can be done, and the search is finished. The search has by definition detected a local optimum, and for each iteration it came closer to the optima. Which is all nice, showing some smartness in the search of a solution. On the other hand, the search is not able to reach the global optima located in (5,5) because of the worsening results in all directions from (3,2).

5.0 Experimental Setup

5.1 Data

The data set used in the study is a set consisting of 33125 matches from four top English divisions in the time period 11.08.2001-13.05.2018, with corresponding odds for each outcome. The last season is kept out of the initial testing and will serve as the final test set when the heuristic has found a local optimum. One thing that should be mentioned is the fact that the odds for a single match is not necessarily taken from the same bookmaker. The best odds for each outcome are picked from `football-datra.co.uk`. Together with which teams playing against each other, the score of the game, and the odds, are there independent variables the author considers most likely to affect the result of a football match based on existing literature and own thoughts.

5.1.1 Dependent Variable

The dependent variable in this research are the result of the matches. The first part of a line of data form the data set is presented below, with all the independent variables following. It is structured the following way: Date, home team, away team, result home team, result away team, odds home win, odds draw, and odds away win.

```
11.02.2018 SOUTHAMPTON LIVERPOOL 0 2 4,38286 3,79571 1,79286
```

In this particular example did Liverpool win 0-2 against Southampton. But for the OLR to understand these results are each outcome translated into the coding system described in the chapter about OLR. For this example, the match would be assigned the value of 3. If Southampton had won, the value assigned would be 1, and a draw would have resulted in the match getting assigned the value 2.

5.1.2 Independent Variables

The independent variables, which is listed and defined below, can be divided into two categories: variables which is used in previous studies, and additional variables new for this thesis. If a variable is denoted with a V do this mean that the variable is used before in previous studies conducted by (Hvattum 2017), (Hvattum, Arntzen, 2010), and (Goddard, 2005), If the notation N is used do this indicate that the variables are first introduced in this study.

V_1 E_{ab} the difference in Elo rating between home team a and away team b before the match is played.

V_2 E_{ab}^{AVG} the average Elo rating of home team and away team before the match is played.

V_3 $E_{ab}^{AVG\text{^}2}$ the square of V_2.

V_4 E_{ab}^2 the square of V_1.

V_5 D_{ab} the natural logarithm of the geographical distance between the home fields for the home and away team.

V_6 $D_{ab}^{<15}$ a binary indicator which is 1 if the travel distance between the two fields is less than 15 km, 0 otherwise.

V_7 I_{ab}^H a binary indicator for the importance of the match from the home team perspective and not for the away team. If the match is important 1, 0 otherwise.

V_8 I_{ab}^A a binary indicator for the importance of the match from the away team perspective and not for the home team. If the match is important 1, 0 otherwise.

V_9 I_{ab} a binary indicator for the importance of the match for both the home and the away team. If the match is not important to any of them 1, 0 otherwise.

V_{10} $G_a^{HS}, G_a^{HC}, G_b^{AS}, G_b^{AC}$ average goals scored and conceded for home and away team.

V_{11} W_{ab} a binary indicator if the match is played on a week-end. 1 if the match is played on a Friday, Saturday, or Sunday, 0 otherwise.

N_{12} $C_a^{HR}, C_a^{HG}, C_b^{AR}, C_b^{AG}$ average corners received and given for home and away team.

N_{13} $F_a^{HR}, F_a^{HG}, F_b^{AR}, F_b^{AG}$ average fouls committed resulting in a free kick. Received and given for home and away team.

5.2 Validation

When determining whether the obtained parameter values are able to predict events from other data than from the sample, i.e. to avoid the model being overfitted, some sort for validation is needed. In this paper a form of k-folding is used. K-folding is a type of cross validation where the data set is divided into k subsets. Each set has the length of n/k, where n is the number of instances in the data set. Then one subset k_i is left for validation, and remaining sets $k_n - k_i$ are used for training. Which k that is used for validation and the remaining that is used for testing will alternate until a the ks have been used for validation. The results from all the validations will then be taken the mean of and further used for statistical investigations. The value of $k = 10$ is used in the thesis, it is a

convenient number and literature has argued that this value yields good results (Yoshua Bengi, Yves Grandvalet, 2004).

For the first run, the $k_1$will be used for validation and $k_2, \dots, k_{10}$ will be used for training. The values of the parameters are then stored, and we continue to the next k, where k_2 now is used for validation and $k_1, k_3, \dots, k_{10}$ is used for training. This procedure continues until we have obtained the values for k_{10}, with $k_1, \dots, k_9$ as training set. For implementation reasons is the method applied in the test somehow different: First the entire data set is shuffled, then the data set is divided into two sub sets. The training set, containing 90 % of the data, and a validation set containing the remaining 10%. After each run the set gets shuffled again. Which in theory implies that the same lines of data almost for sure will occur more than once in the validation set, but from a practical point of view, this does not have to mean so much for the results. Further, as mentioned previously is the last season cut out of the original set, to serve the purpose as a final test set. This set is called a holdout set.

When the whole data set is divided into their respective sub sets will the process of generating the parameter values of the independent variables begin. The OLR uses the training set to generate the parameter values, and the obtained values are used on the validation set with the simulation (see next section). This is then repeated fifty times to get a good sample size for the evaluation of the results. After this procedure is finished, and the values are obtained, the heuristic will target the same problem. Starting with the same parameter values as in the first run with OLR and attempt to tune the parameters in a way that they will yield a higher return. This is also done fifty times, to make comparison with the initial solution found when only relying on the OLR. The overall reason for doing so is to check if the strategy can outperform the OLR and maybe also beat the market. When this is set and done, will the holdout set, which until now is completely unknown to the model, serve as a final test. This to give an impression of the robustness.

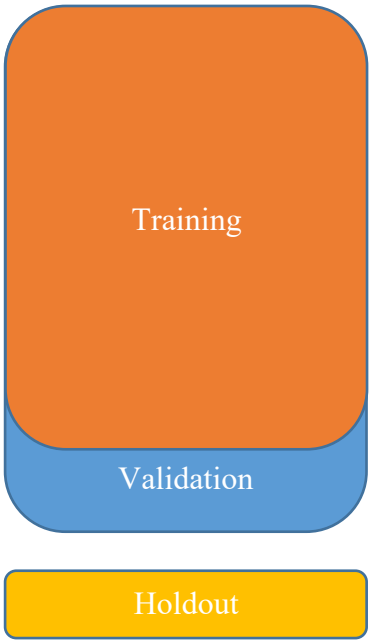

Figure 5.1

5.3 Simulated Betting

When to optimize a problem one has two different possibilities to evaluate the performance of method applied. One can use the objective function, typical $min/max\ f(x) : x \in F$ for linear problems. Even for non-linear problems can the objective function be used as evaluation, as long as the functions are smooth. In such cases as the one we encounter here; this will not be possible because we do not know what the objective function looks like. To be able to evaluate we must therefore introduce a pure evaluation function, typical $h(x)$, which will be the target of maximization.

A betting simulation will be the way to verify if it is possible to detect inefficiency in the odds market. I.e. check if it is possible to develop a betting strategy where the return deviates significantly in a positive direction from the expected return from placing bets blindfolded. The result from the simulation, which is carried out on the validation fold (10% of the matches), will give a return and this value will serve as the evaluation for the performance of the model. The logic is the following: A betting strategy is developed, and it stays consistent throughout the entire process. First when just applying the values of the parameters obtained by using the OLR method, which will serve as our starting point. Then tuning the parameters one by one, checking whether the return on investment (ROI) increases until the function discovers a local optimum. If the betting simulation yields positive return for enough runs, it will be possible to verify a significant difference between marked odds and the estimated probabilities and therefor also to draw a conclusion about the market efficiency.

The betting strategy that will be applied is displayed below. The calculated return is the measurement of how good the heuristic algorithm works and perform. We take advantage of the relationship between odds and probabilities, where the odds are the inverse of the probability. So, if the probability multiplied with the odds are higher than 1 it is understood as the estimated probabilities are higher than the given odds implies, and therefore placing a bet here should be taken into consideration. A threshold is introduced so the positive deviation has to be greater than this before a bet is placed. A lower bound is also introduced.

$P = Probability$
$O = Odds$
$H = Home$
$D = Draw$
$A = Away$
$g = Given$
$c = Calculated$
$\gamma_l = Lower\ bound$

$$\gamma_u = Threshold$$

$$M = max: \{P_{Hc} * O_{Hg}, P_{Dc} * O_{Dg}, P_{Ac} * 0_{Ag}\}$$

$If\ M \geq 1 + \gamma_u\ \&\ \frac{1}{O_{xg}} > \gamma_l^*$ → *Place bet.*

$Invested: Placed\ bets\ *\ Unit$

$$Pay\text{-}outs: \begin{cases} If\ H\ is\ played\ and\ H\ is\ true \\ If\ D\ is\ played\ and\ draw\ is\ true \\ If\ A\ is\ played\ and\ away\ is\ true \end{cases} \rightarrow\ Unit * O_{H,D,A,g}$$

Zero otherwise.

Return= Pay outs-Invested.

** The reason for a lower bound based on the given odds is to possibly remove the games where the calculated probabilities are higher than the given odds imply, but the probability of the given outcome is still (too) low. Let us say a bookmaker estimates the probability of an outcome to be 0.1. Translated to odds* $\frac{1}{0.1} = 10$. *Let us say the heuristic estimates a probability 50 % higher than given, a 15 % chance for the outcome. We get the following calculation.* $0.15 * 10 = 1.5$ *The threshold will most likely be met but placing a bet on something with a 15% probability could lead to a disappointment. It is therefore, in addition to the threshold, introduced a lower bound parameter which are both tuned in the search as the other parameters. For the OLR simulation the parameters are set to* $\gamma_l = 0\ and\ \gamma_u = 1$.

6.0 The Search

There are two searches that are done in the heuristic, which interacts. The first is the tuning of the parameters, second the is the determination of proper threshold and lower bound. The search works the following way: First the search starts by looking at the parameters, when a parameter is changed the simulated betting function will run to calculate a new ROI. Then the bound/threshold search will kick in and find the best bound for the given parameter values. The best bound/threshold values, which is understood as the one that yields the best return, is stored and the search continues. All parameters are tuned until no further improvements can be found. For a given change in parameter value, the bound/threshold search suggest limits that will maximize the return. Which means that a search with and without adjustable limit values will find different solutions.

The sizes of the parameters are unknown to begin with, and it is easy to imagine them having several different values. That is why it could be an idea to increase or decrease the value of the parameters by a value telling something about the parameter itself. The step-length is therefore set to be the standard error of the parameters. Since the value of the thresholds do not have the same attributes as the parameters and cannot be a chosen to a step length as a fraction of itself (lower bound has initial value 0) a fixed step-length is chosen. First try was 0.001 for both bounds, but lower bound seemed to have an issue escaping 0. Second try was 0.05 for lower and 0.001 for upper, where the lower bound was able to escape. Third try was 0.05 for both bounds, which improved the overall results even more. The result was satisfactory enough, and by increasing the step length further one will make the neighborhood to small, and by decreasing one could encounter the issues of the not escaping initial point.

6.1 Pseudo Codes

6.1.1 Pseudo Code for Determining Lower Bound and Threshold.

Compass search for determining limit values

```
Calculate ROI via simulated betting with initial threshold =1 and lower bound =0
While stopping criterion not met do
    For each pair of limit values (0.05,0), (0,-0.05), (-0.05,0) (0,0.05)
      Current threshold+pair(i)
      Calculate ROI with new limit values
     If new ROI > old ROI, update limit values and search with current limit values
     Else
      Continue search with original limit values
    End if
Loop until no further improvements can be done.

 End while
```

6.1.2 Pseudo Code for Tuning β, θ.

Compass search for tuning beta and theta

```
Take parameter values from OLR
Calculate ROI via simulated betting
While stopping criterion not met do
    For each parameter (i)
      parameter(i) + standard error(i)
      Use search for determining limit values
      Calculate ROI with new parameter value and new limit values
     If new ROI > old ROI, update ROI and limit values, go to next parameter value
     Else
      parameter(i) - standard error(i)
      Determaine lower bound and threshold
      Calculate ROI with new parameter value and new limit values
     If new ROI > old ROI, update ROI and go to next parameter value
     Else
       Go to next parameter
    End if
Loop until no further improvements can be done.

 End while
```

6.2 Runs

To test the model performance and in extension the efficiency of the betting market, a simulation must be run. A test in this context means running the simulation. To generate comparable results the shuffling, which is done between the runs, is done by a sequential seed assigning. 50 runs are executed, for the first run seed 10 is selected, second run will get assigned seed 11 and so on. This applies for both the run without and with the compass search. The reason why 50 runs are chosen is to get a decent sample size, and a statistical rule of thumb says a sample size greater than 40 is to be considered a large sample size (Devore, Berk, 2012). To determine whether the results are significantly better than the expected results, a Z-tests is used.

The Z-test is defined the following way:

$$Z = \frac{\bar{x} - \mu_0}{s/\sqrt{n}} \qquad (6)$$

The Z value obtained is then used to find a p-value. A p-value can be reviewed as the value the significance level is compared up against, to determine whether the null hypothesis can be rejected or not. In the tests conducted in this thesis, the relevant investigation will be whether the obtained results are higher than the expected results. And the p-value for one tail where $\bar{x} > \mu_0$ is calculated the following way:

$$P = 1 - \Phi(z) \qquad (7)$$

6.2.1 First Simulation with OLR

The initial testing of the betting simulation was applied with probabilities obtained by only using OLR. This means that the only comparison done is calculated probabilities up against the odds given in the market. The results are displayed in table 1 in the appendix. With no expectancy of being able to out-perform the market at this stage i.e. get positive return, this will only serve as reference point before the compass search is allowed to fine-tune the parameters later on. For each game where the betting criterion is met *one* unit is placed. The validation set consists of 3313 matches, but only the games where the limit-value criteria (for OLR $\gamma_u > 1, \gamma_l = 0$) is met are the bet placed. Before each run is the whole data set shuffled before a new validation fold is created.

As mentioned earlier do the betting companies rely on the overround. For the data set used in this study will, on average, the overround be 3.4%. This can be understood as if one bets blindfolded one should expect to, on average, loose 3,4% on the investment.
A hypothesis test will confirm the visual impression one gets from the results, that there is no basis for claiming that the OLR estimated parameters is able to yield positive return, the more important question is whether it performs significantly better than one could expect for placing bets blindly, a one tailed z-test is used to confirm this. The full result of the simulation can be found in table 1 in the appendix.

In the simulation an average of 2657.34 bets was placed. The rule of the simulation states that $bets\ placed = units$. This could be understood as one, on average, should expect to lose $2657.34\ units * -0.034 = -90.35\ units$ if one placed bets blindfolded. This will be the return the obtained return via simulation will be tested against.

The statistics of interest after the simulations are the following
$Mean\ return = -52.80 \quad Var = 6627.8 \quad SD = 81.411$

The hypotheses are the following:
$H0: return = -90.35 \quad H1: return > -90.35 \quad \alpha = 0.05$

$$Z = \frac{-52.8 + 90.35}{81.411/\sqrt{50}} = 3.26$$

$P - value = 1 - 0.9994 = 0.0006$

Conclusion: The OLR alone performs better than the expected return. And since $\alpha > 0.0006$, the null hypothesis can be discarded. So already when only using the probabilities obtained in OLR it is possible to detect inefficiency in the betting market, but even though the odds are not set correctly are the model unable to beat the margin. i.e. get positive return. The result obtained here is interesting as a reference point for further investigation.

6.2.2 First Simulation with Compass Search

When the compass search is allowed to improve the parameters already generated by OLR, the return is at first glance improves further. But this would be too quick of a conclusion that the results are significantly better than results obtained in OLR. First a test is done to decide whether it performs better than expected return.

The same numbers of runs are conducted with the same seeds as in the simulation done solely based on the parameters from the OLR. The comparison will therefore be based on the exact same data material. The runs can be found in the appendix table 2. Once again is the average expected return of interest, and for the runs done with the guidance of the search 1096.64 bets was placed on average, resulting in an expected return of $1096.65\ units * -0.034 = -37.2861\ units$.

The main statistics in the run was:

$Mean\ return = -15.39 \quad Var = 4590.85 \quad SD = 67.7558$

The hypotheses are the following:

$H0: return = -37.29 \quad H1: return > -37.29 \quad \alpha = 0.05$

$$Z = \frac{-15.39 + 37.2861}{67.7558/\sqrt{50}} = 2.285$$

$P - value = 1 - 0.9887 = 0.0113$

Conclusion: Once again is the obtained return significantly better than expected return. The model, on average, places bets on fewer matches, but the accuracy of the bets placed increases as one can see from the return. Since the number of bets placed has decreased has also the expected return increased. An upwards trend showing is that it is possible to perform better than if one placed bets blindfolded, but still not enough to get a positive average return.

6.2.3 Comparison of OLR and Compass Search

Both OLR and the model with improved parameters found in the compass search are significantly improvements from theirs expected return. But can we say that the compass search is better than the OLR. This is what to be determined in the next test. There is an issue with the different variance between the two different methods. By scaling the expected return to become equal, one neglect the fact that the compass search is stricter by nature and forces it to place bets on imaginary matches it usually would not place bets on.

To determine whether the mean obtained in the compass search differs significantly in a positive direction from the mean obtained in the OLR, an Unequal variance (Welch) t test is used. Defined the following way:

$$T = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \qquad (8)$$

And degrees of freedom v:

$$v = \frac{(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2})^2}{\frac{s_1^4}{{n_1}^2 * n_1 - 1} + \frac{s_2^4}{{n_2}^2 * n_2 - 1}} \qquad (9)$$

When substituting for current values, the T-value and v becomes the following.

$$T = \frac{-15.39 + 52.8}{\sqrt{\frac{4590.85}{50} + \frac{6627.8}{50}}} = 2.50$$

$$v = \frac{(\frac{6627.8}{50} + \frac{4590.85}{50})^2}{\frac{6627.8^2}{50^2 * 49} + \frac{4590.85^2}{50^2 * 49}} \approx 95$$

Giving a value of approximately 0.0075, concluding that results in CS is significantly better than the ones obtained in OLR with $\alpha = 0.05$

6.3 In Search of Lost Information

If one considers the research question is what to be investigated whether it is possible to improve the OLR by detecting some consistent misjudgment the model does when determining the parameter values. To get a better understanding of how the parameter values changes from the initial OLR to after the search, one must also look past the results of the return. To give an impression of how often the search finds it beneficial to adjust the parameters a frequency table is provided. Where throughout the 50 runs it is shown how many times the search makes a change in the parameter value for each variable.

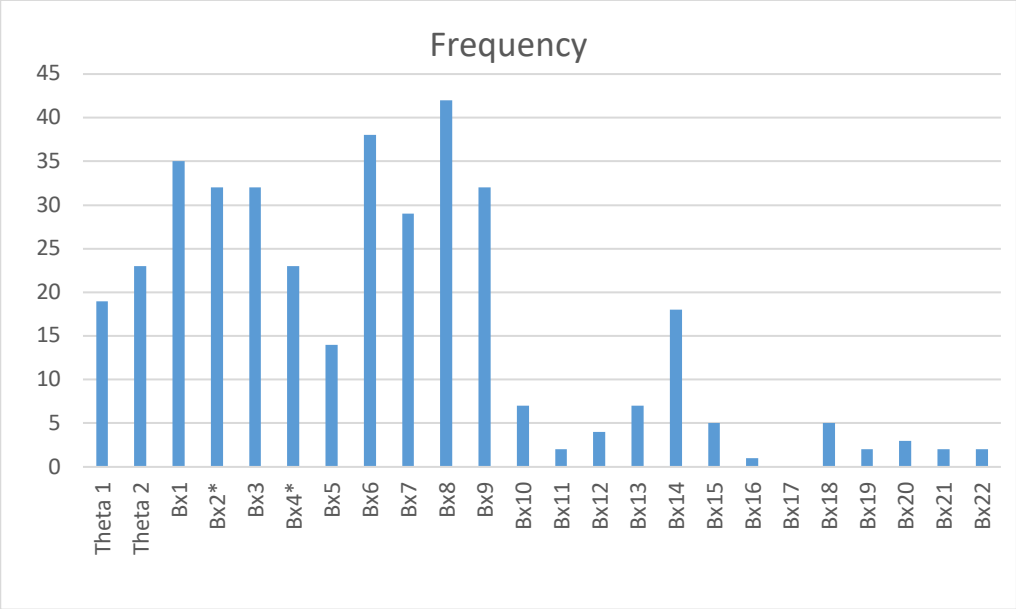

Figure 6.1 *Parameters are changed, but changes are marginal, close to zero, typical less than 0,00001

The frequence table gives us some information: in most of the run, the search finds that tuning the parameter improves the return. But this is somehow not fruitfull on its own. What does the search detect other than the OLR is the question worth envestigating. If one should follow the remarks done in “Ordinal versus nominal regression models and the problem of correctly predicting draws in soccer” (Hvattum 2017) is the most interesting parameters to further investigate the relationship between the adjustment of the *theta* parameters and the variables regarding the importance of the match for the teams. To investigate the relationship between importance of the game for the teams and the distance of the two thetas are a closer investigation done on V7 & V8. V9 also deals with the same issue, but since it describes whether the match is important for both teams or not, it is assumed that it will cancel out each other in the understanding of the distance between theta 1 and theta 2. The main findings of the investigation are the following:

The distance of Theta 1 and Theta 2 is changed 41 times of the 50 runs. To get an understanding of what causes the changes in Theta distance will it also be interesting to look at the changes in V7 and V8. The natural question to ask is which changes are related. If one study the change in one parameter and how this affects the other parameters and again how this affects the return, it could be possible to find a pattern to determine where the OLR fails in greater extent than the compass search, under the assumption that V7 and V8 are the most decisive variables for a draw prediction.

In total there are 11 scenarios which the comparison resulted in. The three main scenarios, which holds 32 of the 50 runs, are the scenario (1) where the parameter value for V7 is decreased more than the parameter value for V8 is increased. The scenario (2) where the parameter value for V7 remains the same, but the parameter value for V8 is increased, and the scenario (3) where the parameter value for V7 is decreased less than the parameter value for V8 increases. For each of the three scenarios the type of change in thetas due to the change in V7 and V8 that occurs most frequently is looked at. It is, unfortunately, not possible to draw any conclusions from the scenarios described above due to lack of consistency. But a comment about the overall trend when comparing the OLR and the CS is that the OLR have a tendency to underestimate the probability for an away win, and therefor also overestimating the probabilities of a draw. In 36 on the 50 runs is the theta distance is decreased. For 28 runs of those 36 runs theta 1 is unchanged. A comparison of returns reveals that CS improved return about 54% of the times.

It also exists runs where the parameter value of V7 and V8 is unchanged, it could therefor be interesting to see if there are changes in theta distance for those runs, and how it affects the return. In total there are 5 runs where the parameters for V7 and V8 are unchanged. The return is increased in all the instances, due to changes in other parameters, or/and stricter limit values. But the greatest impact on the result is located in run 8, where the theta distance has changed significantly.
The full result is displayed in the table below.

Scenarios	Run	Theta 1	Theta 2	Delta Distanc	Return OLR	Compass
Beta7 decreases more than Beta 8 increases	0	Unchanged	Decreases	-0,3263	9,058	107,19
	1	Unchanged	Decreases	-0,3271	-26,20	-43,217
	4	Unchanged	Decreases	-0,3254	-100,34	27,137
	9	Unchanged	Decreases	-0,3263	-172,71	-73,772
	11	Unchanged	Decreases	-0,3264	-53,78	-29,679
	22	Increased	Increased	-6,44E-05	42,20	55,255
	34	Unchanged	Decreases	-0,3259	78,21	50,637
	36	Decreases	Decreases	4,87E-05	-170,16	-111,77
	43	Increased	Unchanged	-0,3272	-59,65	-9,77
	44	Unchanged	Decreases	-3,26E-01	-45,45	-100,332
	45	Unchanged	Decreases	-0,3262	-265,75	-273,867
	46	Decreases	Decreases	-3,26E-01	-176,37	-35,645
Beta 7 is constant, Beta 8 is constant	2	Unchanged	Unchanged	0,00E+00	-68,10	-48,51
	8	Increased	Unchanged	-0,3264	-114,65	29,85
	20	Increased	Unchanged	-3,27E-01	-169,62	-43,74
	38	Increased	Increased	-6,70E-05	-117,56	-1,935
	40	Unchanged	Unchanged	0,00E+00	-14,33	2,343
Beta 7 is constant, Beta 8 increases	5	Unchanged	Unchanged	0	-33,43	59,085
	6	Increased	Increased	-6,00E-05	-103,81	40,35
	7	Increased	Increased	0	-200,53	-28,58
	10	Decreased	Increased	0,292	-27,46	43,511
	14	Unchanged	Decreases	-0,327	-28,00	-45,942
	17	Unchanged	Unchanged	0	10,39	-43,2
	21	Unchanged	Decreases	-0,10185	63,06	28,51
	26	Unchanged	Decreases	-0,32553	3,74	32,222
	29	Unchanged	UNchanged	0	-21,16	-65,71
	33	Increased	Increased	0,325642	-4,49	52,22
	37	Unchanged	Decreases	-0,3257758	-13,75	47,17
	41	Increased	Unchanged	-0,326199	-27,75	29,233

Scenarios	Run	Theta 1	Theta 2	Delta Distanc	Return OLR	Compass
Beta 8 is constant, Beta 7 increases	12	Unchanged	Decreased	-0,326547	10,31	44,25
Beta 8 is constant, Beta 7 decreases	13	Unchanged	Decreases	-0,325003	-9,06	-4,515
	24	Unchanged	Decreases	-0,3266764	34,77	16,915
	48	Unchanged	Decreases	-0,325908	63,99	-12,566
Beta 7 decreases less than Beta 8 increases	15	Unchanged	Decreases	-0,3258166	-19,74	-76,948
	16	Increases	Unchanged	-0,325906	-63,40	25,27
	25	Increases	Unchanged	-0,325764	171,77	57,82
	27	Unchanged	Increases	0,105764	-115,42	-86,448
	28	Unchanged	Decreases	-0,326055	-71,57	-7,057
	31	Unchanged	Decreases	-0,3261865	-82,12	-178,34
	47	Increases	Unchanged	-0,326151	-47,14	4,99
	49	Unchanged	Decreases	-0,326151	35,21	-69,08
Beta 7 decreases more than Beta 8 decreases	18	Unchanged	Decreases	-0,325639	-147,43	-14,988
	30	Unchanged	Decreases	-0,325854	13,91	38,06
Beta 7 increases less than Beta 8 increases	42	Unchanged	Unchanged	0	-61,27	15,196
Beta 7 increases less than Beta 8 decreases	23	Unchanged	Decreases	-0,325578	-85,78	-2,798
Beta 7 is consistant, Beta 8 decreases	19	Unchanged	Unchanged	0	-31,78	-36,114
	32	Unchanged	Decreases	-0,3271885	-63,50	2,69
	35	Decreases	Decreases	7,72E-05	-125,61	-148,604
	39	Unchanged	Unchanged	0	-175,72	-15,71
Beta 7 increases more than Beta 8 increases	3	Unchanged	Decreases	-0,326	-62,29	29,585

Table 6.1

Another interesting aspects to review are the limit values. How much of the improvement is a result of the tuning of the parameters and how much is due to the limit values. To determine

the divisiveness on the impact a new run is done, setting the lower bound to 0 and threshold to 1. The name of the search is Compass Search without Limit values, from now CSwoLv. With the same limit values as in the initial run with the OLR. A tendency is that when limit values are included in the search do the model find it benificial to bet on fewer matches. To give an impression of the different searches operates are the returns together with number of bets placed, for initial run, CSwoLv and standard search with limit values, compared. The results are displayed in the table below.

The main findings in the compairison are that the limits decreases the propensity to place a bet. Where as CSwoLv, only saying that if the estimated probability should be greater than the suggested probability implied by the given odds, acutally ends up placing more bets than the simulation just based on OLR. Since more bets are placed in CSwoLv than in OLR and the average return is better, a conclution that it also performs better than if one had placed bets blindfolded is drawn. The full results can be found in the table below.
To investigate whether the full search outperforms CSwoLv is once again the Welch's t-test applied. Before conducting the test, the variance and mean of the results obtained by CSwoLv is needed: $S_{\text{CSwoLv}} = 8192.1,\ \bar{x}_{\text{CSwoLv}} = -48.08$

Recollect formula (8) and (9) from section 6.2.3.

$$T = \frac{-15.39 + 48.8}{\sqrt{\frac{4590.85}{50} + \frac{8192.1}{50}}} = 2.09$$

$$v = \frac{(\frac{8192.1}{50} + \frac{4590.85}{50})^2}{\frac{8192.1^2}{50^2 * 49} + \frac{4590.85^2}{50^2 * 49}} \approx 91$$

Giving a value of approximately 0.0195

The conclution is that the full search performs significantly better than CSwoLv, and that the limit values helps the search finding more accurate parameter values. Resulting in fewer bets with higher accuracy.

	Initial run.		Without threshold		With thresholds	
Run	Bets placed	Return	Bets placed	Retrun	Bets placed	Return
0	2599	9,058	2704	40,613	1822	107,19
1	2666	-26,20	3117	40,65	1711	-43,217
2	2616	-68,10	3114	-75,001	591	-48,51
3	2638	-62,29	3115	-110,721	1338	29,585
4	2630	-100,34	3113	-84,568	1339	27,137
5	2673	-33,43	2786	0,652999	570	59,085
6	2666	-103,81	3117	45,582	168	40,35
7	2711	-200,53	2854	-123,483	595	-28,58
8	2662	-114,65	2681	-49,208	224	29,85
9	2679	-172,71	3110	-125,346	2076	-73,772
10	2627	-27,46	2745	130,446	2254	43,511
11	2679	-53,78	2758	-29,226	1848	-29,679
12	2645	10,31	3114	103,114	889	44,25
13	2657	-9,06	3116	-54,161	1225	-4,515
14	2667	-28,00	3113	-117,173	1870	-45,942
15	2690	-19,74	3116	-63,259	2191	-76,948
16	2674	-63,40	2739	-95,794	210	25,27
17	2631	10,39	2807	2,441	523	-43,2
18	2650	-147,43	2829	-149,618	1693	-14,988
19	2646	-31,78	2678	37,997	562	-36,114
20	2674	-169,62	2691	-195,047	187	-43,74
21	2659	63,06	2850	77,371	1258	28,51
22	2650	42,20	2824	129,461	138	55,255
23	2652	-85,78	2693	-56,453	882	-2,798
24	2626	34,77	3115	-0,841995	1200	16,915

	Initial run.		Without threshold		With thresholds	
Run	Bets placed	Return	Bets placed	Retrun	Bets placed	Return
25	2641	171,77	2781	71,94	168	57,82
26	2621	3,74	2806	69,575	1111	32,222
27	2663	-115,42	3115	-57,047	2231	-86,448
28	2658	-71,57	3114	-29,834	2004	-7,057
29	2665	-21,16	3115	-21,994	625	-65,71
30	2650	13,91	3116	28,352	841	38,06
31	2681	-82,12	3115	-111,998	1878	-178,34
32	2680	-63,50	3116	-31,669	1169	2,69
33	2677	-4,49	3117	-18,039	134	52,22
34	2664	78,21	3116	21,62	1974	50,637
35	2676	-125,61	3116	-103,59	1380	-148,604
36	2683	-170,16	2704	-182,796	1196	-111,77
37	2628	-13,75	2858	71,976	1229	47,17
38	2686	-117,56	3114	-133,502	145	-1,935
39	2667	-175,72	2836	-151,889	981	-15,71
40	2648	-14,33	3116	-95,281	529	2,343
41	2686	-27,75	2855	-2,523	239	29,233
42	2651	-61,27	2704	-143,322	2232	15,196
43	2659	-59,65	3116	-170,018	237	-9,77
44	2640	-45,45	3114	-58,947	1672	-100,332
45	2669	-265,75	3115	-284,872	1594	-273,867
46	2670	-176,37	2712	-172,258	964	-35,645
47	2639	-47,14	3116	-93,109	257	4,99
48	2662	63,99	3115	-9,45	1310	-12,566
49	2636	35,21	3115	-73,852	1368	-69,08
Avg.	2657,34	-52,80	2962,32	-48,08198	1096,64	-15,38696

Table 6.3

6.4 Initial Final Run

To test the over all robustness of the model one final test is executed. As stated earlier in the thesis is a set named hold-out deliberetly been held out of the model. This set contains the 2017-2018 season for the top four divisions in english football leage, the most current season finished. The final simulation will be tested on this set. Due to the way the training and validation set is shuffled, it could be convinient to try the model on data which has not been availible for the model previously. This is to check whether the model developed can be used as a tool for future predictions. This is usefull for at least two perpouses. Number one is the transferbility to other problems within prediction modelling, the other is of cource the fact that one can use the model to empty the bookmakers pockets. This is also interesting because it revealses the bookmakers development in odds setting accuracy thorughout a season. A theory could be that most of the profit is obtained in the beginning of the season, where the uncertainties are bigger, due to changes which has happened between seasons.

The result of the parameter determination search is displayed below.The initial parameter values obtained by the use of OLR (blue) and the suggested values the compass seacrch thinks the parameters should be set to (orange).

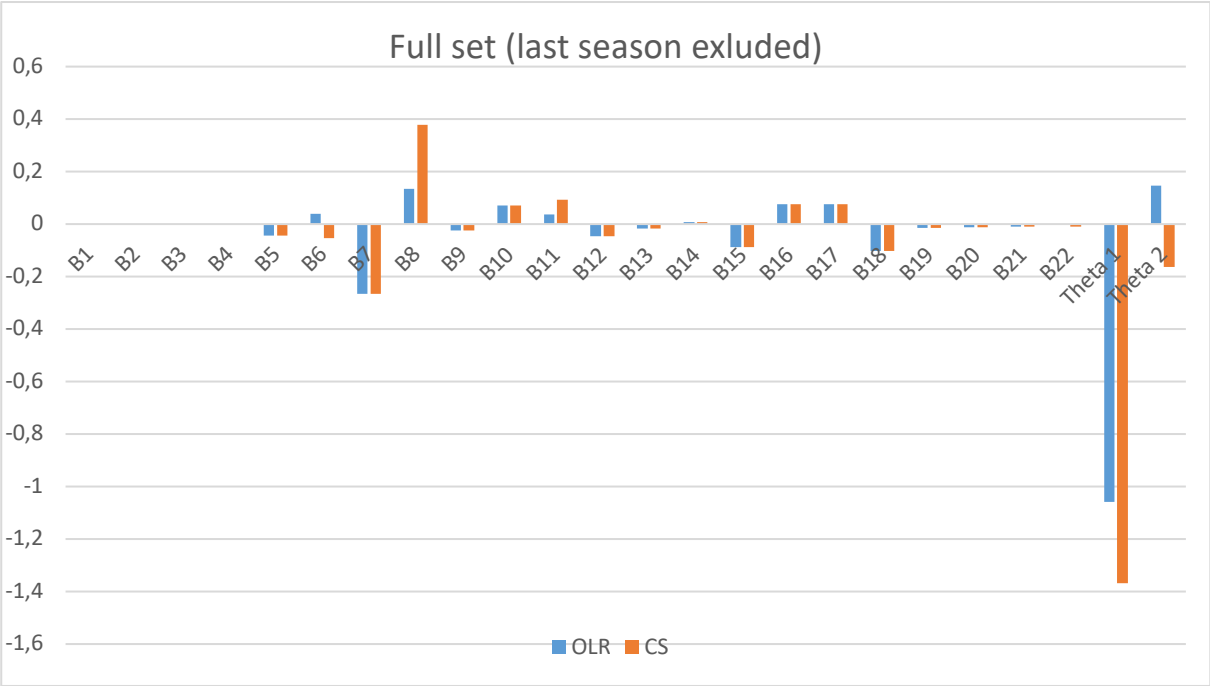

Figure 6.3 Additional relevant information is that the bound and threshold obtained by the use of the compass search was 0.2 for lower bound and for the threshold 1.00. The data is provided in table 3.a in the appendix.

Before the simulation is completed some remarks should be done. Most of the parameters that changes (9 in total) changes marginary, but three stands out: The parameter value of V8, if the match is importaint to the away team but not for home team. The theta 1, which has shifted to the left, decreasing the probability for a home win. Theta 2 has also shifted to the left increasing the probability of an away win. As discussed in section 6.3 does it seems like the OLR has a tendency to over estimate the home ground advantage, neglecting the effect of importance of the match for the away team. If one shold follow (Hvattum, 2017) conclutions would the interesting thing here be to observe the distance between theta 1 and theta 2 and whether it changes. As discussed earlier, a weakness of the OLR model is its disability to decrease or increase the distance between the two theta's. The theta values prior to the search was $-1.05919\ and\ 0.145891$, a distance of 1.205081. After the search the theta had the values $-1.36845\ and\ -0.16332$, a distance of 1.205131. A very small change in width.

The result from the final run is displayed in table 3 b in the appendix. In addition to the standard data is a graph showing the cumulative return over time provided. To caluclate the average expected return on a match in the last season, the overround for the last season is needed. The overround is calculated the same way as before, and for the 2017-2018 season alone, the average overround is 0.925%. Expected retrun for the last season is therefor $1492\ units * -0.00925 = -13.8\ units$. Obtained profit was 5.77 units.

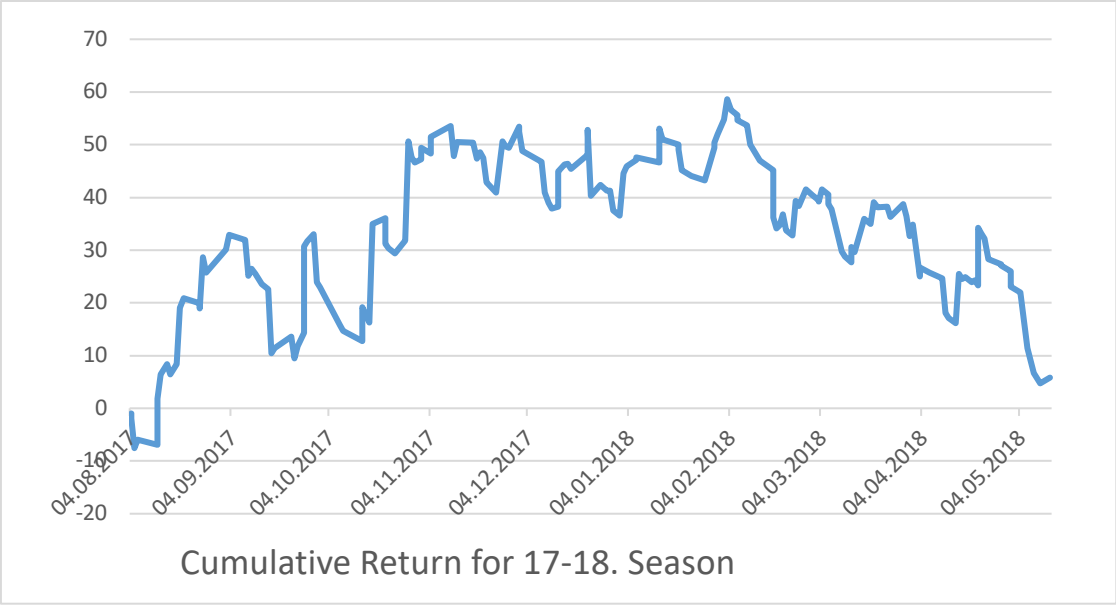

Figure 6.4

Conclution:
The betting stragy seems to out perform the market in the last run. But due to lack of repetitions is it not possible to conclude that the betting strategy performes significantly better than expected return suggest. It shows, however, that it could be possible to beat the market by the use of the strategy created, and that on data which has been unavailable for the model. Another interesting aspect is the development in return and the upwards trend until 04.11.2018 where it stabilizes before the downwards trend starts from around 04.02.2018 and stays throughout the rest of the season. This could imply that bookmakers are more accurate in their odds-setting as the season unfolds, due to the increasment of more current information, which is more valueble than older information.

6.5 Modification

Given the results already obtained would it be interesting to examine several things. First: is it possible to obtain even better solutions? As discussed earlier in the thesis is it common to allow a direct search to change its step length when no further improvements can be done. By changing the step length, one also creates a new neighborhood with the possibility of encounter new solutions which earlier has been unavailable. Second: a set like the one used to conduct the tests is a “best odds” set where the odds for each single outcome for each single match could be taken from different bookmakers. This means that if one should implement the betting strategy developed in this thesis in real attempt to beat the betting market, it would involve a fair amount of time just locating the best odds for each outcome of each match. That is why a new data set with average odds for the same matches is also reviewed, giving odds more realistic to what one can expect from a normal bookmaker, and therefore also checking how the search handles data material where it will be harder to find promising prospects. And least but not last, what happens if the matches containing a “top team” in the sense of popularity is removed as an option to bet on. As discussed previous in the thesis, it could be that bookmakers tend to decrease the odds for those teams, making it impossible for the search to place a bet on those teams given the treading strategy.

6.5.1 Alternative Step Length.

As discussed earlier in this theisis one can expect the true parameter value lie within two standard errors from the estimated parameter value. With this in mind could it be interesting to halfen the step length, $std.err/2$. When allowning the step length to be halfen two

scenarios are possible. First of all, one will create a new neighborhood of solutions and by doing so double the chance for hitting the true parameter value. But as shown in figure *4.2,* a too small step length can result in the model not being able to escape a solution even if there exist better solutions in the solution space. Therefore are two different runs tested. One where the step length is $std.err/2$ from start to end, and one where the the step length is decreast only after the stoping criterion, no further improvements, is met.. The search with fixed step length "parameter error(i)" is used as a referance.

	Compass		Half step throughout		Half step dynamic	
	Bets	Return	Bets	Return	Bets	Return
	168	57,82	282	33,23	237	71,59
	1111	32,222	1085	23,512	1559	18,055
	2231	-86,448	1821	-97,288	2076	-79,389
	2004	-7,057	518	-15,347	532	-26,807
	625	-65,71	607	-78,29	628	-33,99
	841	38,06	530	14,17	852	44,65
	1878	-178,34	1554	-166,203	1932	-170,797
	1169	2,69	1193	-5,59	1658	11,746
	134	52,22	1222	-21,122	142	45,22
	1974	50,637	534	56	1926	51,867
	1380	-148,604	1866	-64,668	1493	-134,714
	1196	-111,77	2023	-90,747	1215	-102,183
	1229	47,17	1159	19,317	1148	58,34
	145	-1,935	361	-24,535	1479	-14,146
	981	-15,71	1884	-103,054	1213	-36,51
	529	2,343	528	-16,227	532	-1,047
	239	29,233	2045	-46,733	129	36,16
	2232	15,196	321	33,19	2194	-7,544
	237	-9,77	1897	-62,534	234	-3,49
	1672	-100,332	2084	-120,012	1671	-86,452
	1594	-273,867	2192	-251,313	1532	-196,127
	964	-35,645	981	-43,245	976	-59,355
	257	4,99	379	-37,43	293	5,43
	1310	-12,566	1144	-7,77	1178	-28,976
	1368	-69,08	539	-14,795	1322	-42,23
Avg.	1096,64	-15,38696	1027,04	-26,96502002	1085,06	-16,0633

Table 6.4

When step length is halved thorughout the intire run do the search performe poorer than with initial step length. This could be due to the weakness of getting stuck in a less fruitfull area of the soulton space. With dynamic step length is the results almost equal both in number of runs and return. It is therfore not possible to conclude that dynamic change in step-length will improve the search, on the other hand it is not possible to conclude that it will not improve the search.

6.5.2 Alternative Data Set.

To test the model on more realistic data is another sets investigated. The data is now average odds instead of max odds as used when conducting the initial simulation.
The average overround for the intire set is 8,72%. Since the overround is greater, the expected return also will be lower per bet, but since the odds now are lower will the search also detect fewer oportunities to place the bet. By the observations done of the privous tests is the search with the "decreasing step-length after no improvements can be done" search chosen. Even though it did not yield a better result than the search without changing step length, it should, in theory, be able to look at more solutions, and therefore also maybe finding an even better one.

The full results can be found in table 4 in the appendix. Once again is a Z-test conducted to investigate wheter the search is able to perform better than expected return. Expected return is now $106\ units * -0.0872 = -9.2432$

The main statistics in the run was:

$Mean\ return\ = -3.44 \qquad Var\ = 189.03 \qquad \mathrm{SD} = 13.749$

Giving the following hypotheses:

$H0:\ return\ = -9.2432 \qquad H1: return > -9.2432 \qquad \alpha = 0.05$

$$Z = \frac{-3.44 + 9.2432}{13.749/\sqrt{50}} = 2.98$$

$$P - value\ = 1 - 0.9986 = \ 0.0014$$

Once again is the observed retrun significantly better than the expected return, but fails to beat the market.

6.5.3 Disregarding Betting on "Favourites"

As discussed in section *3.5* could it be interesting to check what happens if one remove the possibility to bet on matches where a top team in terms of popularity is attending. In the simulation for the validation fold a new constrtaint is created, namley not allowed to place bets on matches where "MAN UNITED" or "LIVERPOOL" plays either as the home or the away team. The runs shows interesting results and for the first time a positive average return is descovered. The reason for this could be that if the bookmakers, deliberately, lowers the odds on the popular teams will this also mean that the bookmaker suggests that the probability of a win is higher for those teams. The search could therefor detect an opportunity of betting against the popular team, leading to a loss. The full results can be found in table 5 in the appendix. Due to the result is it also, for the first time, possible to test the hypothesis whether the return is significantly better than zero.

$Mean\,return\ = 16.13 \qquad Var\ = 4240.8 \qquad SD\ = 65.12$

Giving the following hypotheses:

$H0:\ return\ = 0 \qquad H1: return > 0 \qquad \alpha = 0.05$

$$Z = \frac{16.13}{65.12/\sqrt{50}} = 1.73$$

$P - value\ = 1 - 0.9582 = 0.0418$

Since P-value $< \alpha$, is the conclution that by removing Liverpool and Manchester United from the search, the search are able to, on average, obtain a result significantly higher than zero.

6.5.4 Final Final Run

When taking all the information obtained during the thesis into consideration a final test is conducted, first and foremost to check how good of a result it is possible to obtain. We have seen that we obtained positive results for the run with the hold out set, we have also seen that when removing the "favorites" the average return was positive. We have seen that dynamic change in step length is not inferior to constant. The final search will therefore be performed on the hold out set, with both initial and dynamic step length, and all the matches containing the most commercial recognized clubs (Manchester United and Liverpool) are tabu to bet on.

Since there is no shuffling involved in the hold out set can one compare the results one on one with the results obtained in the initial final run from section 6.4.
A comparison of the three runs shows the following:

	Bets placed	Return	Threshold	Lower bound	Expected return
Initial	1492	5,77	1	0,2	-13,801
Fixed step length	1470	27,77	1	0,2	-13,5975
Dynamic	782	87,71	1,15	0,25	-7,2335

Table 6.5

The results support the findings in section 6.5.3. It also shows that a change in step length, when no further improvements can be done, could be helpful when navigating through the solution space in search for an even better solution.

7.0 Concluding Remarks

In this master thesis, it is shown that both the OLR and the CS in various forms are able to perform significantly better than expected return suggests in simulated betting. And due to this fact, it is possible to conclude that the bookmakers are not able to set odds completely correct i.e. the betting market for football in England is inefficient. Even though an inefficiency is detected, the search failed to profit from the inefficiency, meaning that the bookmakers overround is higher than their misjudgment in odds setting. It is also shown that a direct search such as CS on top of OLR is able to outperform OLR alone. The research has not succeeded in finding a clear tendency in why this is the case, but it seems like OLR tends to overestimate the probability of draws.

The only instance where the search was able to obtain an average return higher than zero was when all the matches containing either Manchester United or Liverpool were removed as an option to place a bet on. This could confirm that the bookmakers purposely lower the odds for the top teams due to the common man's belief in those teams, disregarded the actual probability. The findings suggest that further investigation with more advanced search algorithms and with other leagues is worth its while.

8.0 Acknowledgements

The author is grateful for the help provided by supervisor Lars Magnus Hvattum. The accessibility he has shown throughout the thesis is highly appreciated. The author would also like to thank Erik Hulleberg for insightful discussions about football related questions.

References

W. Andreff, S. Szymanski, *Handbook of the Economics of Sports*, (2006) 41–43.

Y. Bengi, Y. Grandvalet, No Unbiased Estimator of the Variance of K-Fold Cross-Validation, *Journal of Machine Learning Research 5* (2004) 1089–1105.

J. L. Devore, K. N. Berk, *Modern Mathematical Statistics with Applications* (2012)

A. E. Elo, The new U.S.C.F. rating system, *Chess Life 16* (1961) 160–161.

D. Forrest, J. Goddard, R. Simmons, Odds-Setters as Forecasters: The Case of English Football, *International Journal of Forecasting 21(3)* (2005) 551-564.

Gambling Commission, *Industry Statistics* (2018) read 01.05.19 https://www.gamblingcommission.gov.uk/PDF/survey-data/Gambling-industry-statistics.pdf

G. Gigerenzer, C. Engel (Eds.), Heuristics and the law, Cambridge, MA: *MIT Press.* (2006) 17–44.

M. E. Glickman, A. C. Jones, Rating the Chess Rating System (1999)

J. Goddard, I. Asimakopoulos. Forecasting football results and the efficiency of fixed-odds betting. *Journal of Forecasting 23* (2004) 51–66.

J. Gross, L. Rebeggiani, Chance or Ability? The Efficiency of the Football Betting Market Revisited (2018)

J. Haigh, *Taking chances* chapter 11 (1999)

L.M. Hvattum, H.Arntzen, Using ELO ratings for match results predictions in association football, *International Journal of forecasting 26* (2010) 460–470.

L.M. Hvattum, F. Glover, Finding local optima of high-dimensional functions using direct search methods*, European Journal of Operational Research 195* (2009) 31–45.

L.M. Hvattum, Ordinal versus nominal regression models and the problem of correctly predicting draws in soccer*, International Journal of Computer Science in Sport Volume 16, Issue 1* (2017) 50–64.

S. Poshakwale, Evidence on Weak Form Efficiency and Day of the Week Effect in the Indian Stock Market, *FINANCE INDIA Vol. X No. 3*, (1996) 605-616.

R. D. Sauer, The Economics of Wagering Markets. *Journal of Economic Literature 36(4)* (1998) 2021–2064.

Appendix

Tables

Table 1. (Results from 50 runs with OLR estimated parameters)

Run	Seed	Bets placed	Return	Exp. return	Return > Exp. Return
0	10	2599	9,06	-88,37	97,42
1	11	2666	-26,20	-90,64	64,44
2	12	2616	-68,10	-88,94	20,84
3	13	2638	-62,29	-89,69	27,41
4	14	2630	-100,34	-89,42	-10,92
5	15	2673	-33,43	-90,88	57,45
6	16	2666	-103,81	-90,64	-13,16
7	17	2711	-200,53	-92,17	-108,35
8	18	2662	-114,65	-90,51	-24,14
9	19	2679	-172,71	-91,09	-81,62
10	20	2627	-27,46	-89,32	61,86
11	21	2679	-53,78	-91,09	37,31
12	22	2645	10,31	-89,93	100,24
13	23	2657	-9,06	-90,34	81,28
14	24	2667	-28,00	-90,68	62,68
15	25	2690	-19,74	-91,46	71,72
16	26	2674	-63,40	-90,92	27,52
17	27	2631	10,39	-89,45	99,84
18	28	2650	-147,43	-90,10	-57,33
19	29	2646	-31,78	-89,96	58,18
20	30	2674	-169,62	-90,92	-78,71
21	31	2659	63,06	-90,41	153,47
22	32	2650	42,20	-90,10	132,30
23	33	2652	-85,78	-90,17	4,39
24	34	2626	34,77	-89,28	124,06

Return>Exp. Value 36

Run	Seed	Bets placed	Return	Exp. return	Return > Exp. Return
25	35	2641	171,77	-89,79	261,56
26	36	2621	3,74	-89,11	92,86
27	37	2663	-115,42	-90,54	-24,87
28	38	2658	-71,57	-90,37	18,80
29	39	2665	-21,16	-90,61	69,45
30	40	2650	13,91	-90,10	104,01
31	41	2681	-82,12	-91,15	9,04
32	42	2680	-63,50	-91,12	27,62
33	43	2677	-4,49	-91,02	86,53
34	44	2664	78,21	-90,58	168,79
35	45	2676	-125,61	-90,98	-34,63
36	46	2683	-170,16	-91,22	-78,93
37	47	2628	-13,75	-89,35	75,60
38	48	2686	-117,56	-91,32	-26,24
39	49	2667	-175,72	-90,68	-85,04
40	50	2648	-14,33	-90,03	75,71
41	51	2686	-27,75	-91,32	63,58
42	52	2651	-61,27	-90,13	28,87
43	53	2659	-59,65	-90,41	30,76
44	54	2640	-45,45	-89,76	44,31
45	55	2669	-265,75	-90,75	-175,00
46	56	2670	-176,37	-90,78	-85,59
47	57	2639	-47,14	-89,73	42,59
48	58	2662	63,99	-90,51	154,49
49	59	2636	35,21	-89,62	124,83

Table 2. (Results from 50 runs with compass-search improved parameters)

Run	Seed	Bets placed	Threshold	Lower bound	Retrun	Exp. return	Return > Exp. Return
0	10	1822	1,15	0,2	107,19	-61,948	169,138
1	11	1711	1,15	0,2	-43,217	-58,174	14,957
2	12	591	1,2	0	-48,51	-20,094	-28,416
3	13	1338	1,15	0,25	29,585	-45,492	75,077
4	14	1339	1,15	0,25	27,137	-45,526	72,663
5	15	570	1,2	0,05	59,085	-19,38	78,465
6	16	168	1,4	0,05	40,35	-5,712	46,062
7	17	595	1,2	0	-28,58	-20,23	-8,35
8	18	224	1,45	7,45E-09	29,85	-7,616	37,466
9	19	2076	1,1	0,2	-73,772	-70,584	-3,188
10	20	2254	1	0,2	43,511	-76,636	120,147
11	21	1848	1,15	0,2	-29,679	-62,832	33,153
12	22	889	1,2	0,25	44,25	-30,226	74,476
13	23	1225	1,25	0,2	-4,515	-41,65	37,135
14	24	1870	1,15	0,2	-45,942	-63,58	17,638
15	25	2191	1,1	0,2	-76,948	-74,494	-2,454
16	26	210	1,45	7,45E-09	25,27	-7,14	32,41
17	27	523	1,2	0,1	-43,2	-17,782	-25,418
18	28	1693	1,2	0,2	-14,988	-57,562	42,574
19	29	562	1,2	0,1	-36,114	-19,108	-17,006
20	30	187	1,45	0,05	-43,74	-6,358	-37,382
21	31	1258	1,15	0,25	28,51	-42,772	71,282
22	32	138	1,45	0	55,255	-4,692	59,947
23	33	882	1,25	0,25	-2,798	-29,988	27,19
24	34	1200	1,15	0,25	16,915	-40,8	57,715

Run	Seed	Bets placed	Threshold	Lower bound	Retrun	Exp. return	Return > Exp. Return
25	35	168	1,5	0,05	57,82	-5,712	63,532
26	36	1111	1,2	0,25	32,222	-37,774	69,996
27	37	2231	1,1	0,2	-86,448	-75,854	-10,594
28	38	2004	1,15	0,2	-7,057	-68,136	61,079
29	39	625	1,2	0	-65,71	-21,25	-44,46
30	40	841	1,25	0,25	38,06	-28,594	66,654
31	41	1878	1,15	0,2	-178,34	-63,852	-114,488
32	42	1169	1,15	0,25	2,69	-39,746	42,436
33	43	134	1,65	0,05	52,22	-4,556	56,776
34	44	1974	1,15	0,2	50,637	-67,116	117,753
35	45	1380	1,25	0,2	-148,604	-46,92	-101,684
36	46	1196	1,1	0,25	-111,77	-40,664	-71,106
37	47	1229	1,15	0,25	47,17	-41,786	88,956
38	48	145	1,45	0,05	-1,935	-4,93	2,995
39	49	981	1,1	0,25	-15,71	-33,354	17,644
40	50	529	1,2	0,1	2,343	-17,986	20,329
41	51	239	1,45	7,45E-09	29,233	-8,126	37,359
42	52	2232	1	0,2	15,196	-75,888	91,084
43	53	237	1,45	0	-9,77	-8,058	-1,712
44	54	1672	1,2	0,2	-100,332	-56,848	-43,484
45	55	1594	1,25	0,05	-273,867	-54,196	-219,671
46	56	964	1,2	0,25	-35,645	-32,776	-2,869
47	57	257	1,45	0	4,99	-8,738	13,728
48	58	1310	1,15	0,25	-12,566	-44,54	31,974
49	59	1368	1,15	0,25	-69,08	-46,512	-22,568

Table 3.a. (Parameter pre and post initial final run (hold out set excluded))

Varaible	B1	B2	B3	B4	B5	B6	B7	B8
OLR	**-0,0033966**	**-1,45126E-07**	**-0,0002277**	**1,20E-09**	**-0,0446568**	**0,0398168**	**-0,2647**	**0,135189**
CS	**-0,00318**	**-1,98E-06**	**-0,0002848**	**1,20E-09**	**-0,0446568**	**-0,0543455**	**-0,2647**	**0,377798**
Diff	0,0002166	-1,83772E-06	-5,711E-05	0	0	-0,0941623	0	0,242609
Varaible	B9	B10	B11	B12	B13	B14	B15	B16
OLR	**-0,0233903**	**0,0720387**	**0,0371906**	**-0,0464843**	**-0,0164891**	**0,00671014**	**-0,08669**	**0,0756207**
CS	**-0,0233903**	**0,0720387**	**0,0933421**	**-0,0464843**	**-0,0164891**	**0,00671014**	**-0,08669**	**0,0756207**
Diff	0	0	0,0561515	0	0	0	0	0
Varaible	B17	B18	B19	B20	B21	B22	Theta 1	Theta 2
OLR	**0,0752152**	**-0,102745**	**-0,0137222**	**-0,0115864**	**-0,0104821**	**-8,128E-05**	**-1,05919**	**0,145891**
CS	**0,0752152**	**-0,102745**	**-0,0137222**	**-0,0115864**	**-0,0104821**	**-0,0104963**	**-1,36845**	**-0,16332**
Diff	0	0	0	0	0	-0,010415	-0,30926	-0,309211

Table 3.b.

Bets placed	Return	Threshold	Lower bound	Expected return
1492	5,77	1	0,2	-13,801

Table 4 (Compass search on data set with mean odds)

Run	Seed	Bets	Return	UT 0.05	LB 0.05
0	10	101	-12,2035	1,5	0,1
1	11	83	23,1547	1,6	0,1
2	12	128	9,43935	1,35	0,1
3	13	63	-12,7954	1,15	0,3
4	14	73	11,3213	1,35	0,25
5	15	162	-16,4771	1,3	0,25
6	16	50	-11,9193	1,75	0,25
7	17	123	-3,3599	1,35	0,2
8	18	71	7,5606	1,4	0,2
9	19	81	-19,9512	1,6	0,1
10	20	41	-3,24829	1,25	0,2
11	21	76	3,19593	1,25	0,2
12	22	84	14,6878	1,45	0,2
13	23	147	6,10141	1,15	0,25
14	24	208	-17,0581	1,25	0,25
15	25	117	-21,9589	1,25	0,25
16	26	84	1,12468	1,25	0,25
17	27	117	-21,7688	1,3	0,25
18	28	98	6,55976	1,5	0,1
19	29	59	2,8227	1,4	0,25
20	30	89	-13,5753	1,45	0,1
21	31	138	-1,29528	1,4	0,1
22	32	59	22,0347	1,7	0,25
23	33	126	-16,7584	1,5	0,1
24	34	126	-11,123	1,5	0,1

Run	Seed	Bets	Return	UT 0.05	LB 0.05
25	35	102	21,4441	1,5	0,1
26	36	122	-12,8766	1,4	0,25
27	37	79	-28,2813	1,4	0,2
28	38	221	-9,4545	1,15	0,25
29	39	85	15,2868	1,6	0,1
30	40	79	-17,656	1,35	0,1
31	41	151	-16,6692	1,3	0,25
32	42	162	-10,0111	1,25	0,25
33	43	28	12,1497	1,5	0,3
34	44	147	3,71284	1,4	0,25
35	45	55	10,2281	1,4	0,25
36	46	162	7,71652	1,15	0,25
37	47	172	-15,246	1,25	0,25
38	48	89	19,8687	1,15	0,25
39	49	77	2,34478	1,55	0,1
40	50	71	-9,89705	1,4	0,2
41	51	160	5,25914	1,25	0,25
42	52	75	-19,1795	1,6	0,2
43	53	109	-6,00402	1,5	0,1
44	54	71	-26,3477	1,4	0,25
45	55	87	-5,37248	1,45	0,1
46	56	134	-0,67968	1,4	0,2
47	57	52	-8,84889	1,45	0,2
48	58	151	-23,4151	1,3	0,25
49	59	139	6,67515	1,35	0,1

Table 5. (Result of simulation when possibility of betting on matches containing popular teams is removed)

Run	Seed	Bets placed	Return	Threshold	LB
0	10	1778	151,19	1,15	0,2
1	11	1684	-16,217	1,15	0,2
2	12	557	-14,51	1,2	0
3	13	1296	71,585	1,15	0,25
4	14	1300	66,137	1,15	0,25
5	15	539	90,085	1,2	0,05
6	16	142	66,35	1,4	0,05
7	17	561	5,42	1,2	0
8	18	192	61,85	1,45	7,45E-09
9	19	2037	-34,772	1,1	0,2
10	20	2192	105,511	1	0,2
11	21	1812	6,321	1,15	0,2
12	22	879	54,25	1,2	0,25
13	23	1208	12,485	1,25	0,2
14	24	1831	-6,942	1,15	0,2
15	25	2146	-31,948	1,1	0,2
16	26	184	51,27	1,45	7,45E-09
17	27	508	-28,2	1,2	0,1
18	28	1654	24,012	1,2	0,2
19	29	536	-10,114	1,2	0,1
20	30	169	-25,74	1,45	0,05
21	31	1234	52,51	1,15	0,25
22	32	125	68,255	1,45	0
23	33	875	4,202	1,25	0,25
24	34	1180	36,915	1,15	0,25

Run	Seed	Bets placed	Return	Threshold	LB
25	35	149	76,82	1,5	0,05
26	36	1079	64,222	1,2	0,25
27	37	2177	-32,448	1,1	0,2
28	38	1942	54,943	1,15	0,2
29	39	585	-25,71	1,2	0
30	40	826	53,06	1,25	0,25
31	41	1836	-136,34	1,15	0,2
32	42	1143	28,69	1,15	0,25
33	43	115	71,22	1,65	0,05
34	44	1925	99,637	1,15	0,2
35	45	1355	-123,604	1,25	0,2
36	46	1179	-94,77	1,1	0,25
37	47	1199	77,17	1,15	0,25
38	48	123	20,065	1,45	0,05
39	49	961	4,29	1,1	0,25
40	50	516	15,343	1,2	0,1
41	51	215	53,233	1,45	7,45E-09
42	52	2179	68,196	1	0,2
43	53	203	24,23	1,45	0
44	54	1623	-51,332	1,2	0,2
45	55	1531	-210,867	1,25	0,05
46	56	945	-16,645	1,2	0,25
47	57	228	33,99	1,45	0
48	58	1269	28,434	1,15	0,25
49	59	1334	-35,08	1,15	0,25
		1065,12	16,13304	1,231	0,152

Code

Code for initial run with parameters obtained with OLR:

```
// BettingAnlalysis.hpp
// Master Thesis
//
// Created by Hans Jacob Brun on 25/01/2019.
// Copyright © 2019 Hans Jacob Brun. All rights reserved.
//

#ifndef BettingAnalysis_hpp
#define BettingAnalysis_hpp

#include <stdio.h>
#include <vector>
#include <iostream>
#include <fstream>
#include <sstream>
#include <cstdlib>
#include <algorithm>

using namespace std;

struct date {
    int day;
    int month;
    int year;
};

struct match {
    int ID;
    date date;
    string home;
    string away;
    int result[2];
    int y;
    vector<double> probs;
    float odds[3];
    float elodif;
    float elodif2;
    float elomean;
    float elomean2;
    float distlog;
```

```
    bool lessthan15km;
    bool importance[3];
    float homegoals[2];
    float awaygoals[2];
    bool weekendmatch;
    float cornerhome[2];
    float corneraway[2];
    float freekickhome[2];
    float freekickaway[2];

    match() : probs(3) {}
};

class BettingAnalysis {
    //Variables:
protected:
    int matchnumber;
    string filename;
    match* data;
    int seed;
    int* indices;
    int K;
    int Ksize;
    //Functions:
public:
    match getMatch(int index);
    void extractKsets(match** Ksets);
protected:
    void loadData();
    void calculate_y();
    void shuffleIndices();
    //Constructor:
public:
    BettingAnalysis(string fname,int mnum,int Knum);
    //Destructor:
public:
    ~BettingAnalysis();
};

#endif /* BettingAnlalysis_hpp */



//  BettingAnlalysis.cpp
//  Master Thesis
```

```
//
//  Created by Hans Jacob Brun on 25/01/2019.
//  Copyright © 2019 Hans Jacob Brun. All rights reserved.
//

#include "BettingAnalysis.hpp"


using namespace std;

match BettingAnalysis::getMatch(int index) {
    if (index < matchnumber) {
        return data[index];
    } else {
        cout << "Could not load match " << index << " from data." << endl;
        match nullmatch;
        return nullmatch;
    }

}


void BettingAnalysis::extractKsets(match** Ksets) {
    shuffleIndices();
    for (int i = Ksize; i < matchnumber; i++) {
        Ksets[0][i-Ksize] = data[indices[i]];
    }
    for (int i = 0; i < Ksize; i++) {
        Ksets[1][i] = data[indices[i]];
    }
}




void BettingAnalysis::loadData() {
    ifstream fin;
    string line;
    fin.open(filename);
    if (fin.fail()) {
        cout << "Could not load data from '" << filename << "'." << endl;
        return;
```

```
}
for (int i = 0; i < matchnumber; i++) {

    data[i].ID = i;

    getline(fin,line,'/');
    data[i].date.day = stoi(line);
    getline(fin,line,'/');
    data[i].date.month = stoi(line);
    getline(fin,line,',');
    data[i].date.year = stoi(line);

    getline(fin,line,',');
    data[i].home = line;
    getline(fin,line,',');
    data[i].away = line;

    getline(fin,line,',');
    data[i].result[0] = stoi(line);
    getline(fin,line,',');
    data[i].result[1] = stoi(line);

    getline(fin,line,',');
    if (line.empty()) {
        data[i].odds[0] = -1;
    } else {
        data[i].odds[0] = stof(line);
    }
    getline(fin,line,',');
    if (line.empty()) {
        data[i].odds[1] = -1;
    } else {
        data[i].odds[1] = stof(line);
    }
    getline(fin,line,',');
    if (line.empty()) {
        data[i].odds[2] = -1;
    } else {
        data[i].odds[2] = stof(line);
    }

    getline(fin,line,',');
    data[i].elodif = stof(line);
    getline(fin,line,',');
    data[i].elomean = stof(line);
```

```
getline(fin,line,',');
data[i].elomean2 = stof(line);
getline(fin,line,',');
data[i].elodif2 = stof(line);

getline(fin,line,',');
data[i].distlog = stof(line);
getline(fin,line,',');
data[i].lessthan15km = stoi(line);

getline(fin,line,',');
data[i].importance[0] = stoi(line);
getline(fin,line,',');
data[i].importance[1] = stoi(line);
getline(fin,line,',');
data[i].importance[2] = stoi(line);

getline(fin,line,',');
data[i].homegoals[0] = stof(line); //sluppet inn hjemmelag
getline(fin,line,',');
data[i].homegoals[1] = stof(line); //scoret hjemmelag
getline(fin,line,',');
data[i].awaygoals[0] = stof(line); //sluppet inn bortelag
getline(fin,line,',');
data[i].awaygoals[1] = stof(line); //scoret bortelag

getline(fin,line,',');
data[i].weekendmatch = stoi(line);

getline(fin,line,',');
data[i].cornerhome[0] = stof(line); //corner vunnet av hjemmelag
getline(fin,line,',');
data[i].cornerhome[1] = stof(line); //corner avgitt av hjemmelag
getline(fin,line,',');
data[i].corneraway[0] = stof(line); //corner vunnet av bortelag
getline(fin,line,',');
data[i].corneraway[1] = stof(line); //corner avgitt av bortelag

getline(fin,line,',');
data[i].freekickhome[0] = stof(line); //frispark vunnet av hjemmelaget
getline(fin,line,',');
data[i].freekickhome[1] = stof(line); //frispark avgitt av hjemmelaget
getline(fin,line,',');
data[i].freekickaway[0] = stof(line); //frispark vunnet av bortelaget
```

```
        getline(fin,line,'\n');
        data[i].freekickaway[1] = stof(line); //frispark avgitt av bortelaget

    }
    fin.close();
}

void BettingAnalysis::calculate_y(){
    for (int i = 0; i < matchnumber; i++){
        if (data[i].result[0] > data[i].result[1]) {
            data[i].y = 1;
        } else if (data[i].result[0] == data[i].result[1]) {
            data[i].y = 2;
        } else {
            data[i].y = 3;
        }
    }
}

void BettingAnalysis::shuffleIndices() {
    for (int i = 0; i < matchnumber; i++) {
        int j = rand() % matchnumber;
        swap(indices[i],indices[j]);
    }
}

BettingAnalysis::BettingAnalysis(string fname, int mnum, int Knum) {
    filename=fname;
    matchnumber=mnum;
    K = Knum;
    Ksize = matchnumber/K;
    data = new match[matchnumber];
    loadData();
    calculate_y();
    indices = new int[matchnumber];
    for (int i = 0; i < matchnumber; i++) {
        indices[i] = i;
    }
    seed = 55;
    srand(seed);
}

BettingAnalysis::~BettingAnalysis() {
    delete[] data;
    delete[] indices;
```

```
}



//  main.cpp
//  Master Thesis
//
//  Created by Hans Jacob Brun on 25/01/2019.
//  Copyright © 2019 Hans Jacob Brun. All rights reserved.
//

#include "BettingAnalysis.hpp"

#include <iostream>
#include <vector>
#include "OLR.hpp"
#include <algorithm>



using namespace std;

int main () {

    int datasize = 33124;
    int K = 10;
    int Ksize = datasize/K;

    BettingAnalysis Analysis1("Masterdata.txt",datasize,K);

    match** Ksets = new match*[2];
    //Training sample
    Ksets[0] = new match[datasize-Ksize];
    //Validation sample
    Ksets[1] = new match[Ksize];

    Analysis1.extractKsets(Ksets);

    vector<vector<double>> trainingX(datasize-Ksize,vector<double>(22));
    vector<int> trainingY(datasize-Ksize);
    for (int i = 0; i < datasize-Ksize; i++){
        trainingX[i][0] = Ksets[0][i].elodif;
        trainingX[i][1] = Ksets[0][i].elodif2;
        trainingX[i][2] = Ksets[0][i].elomean;
        trainingX[i][3] = Ksets[0][i].elomean2;
```

```
    trainingX[i][4] = Ksets[0][i].distlog;
    trainingX[i][5] = Ksets[0][i].lessthan15km;
    trainingX[i][6] = Ksets[0][i].importance[0];
    trainingX[i][7] = Ksets[0][i].importance[1];
    trainingX[i][8] = Ksets[0][i].importance[2];
    trainingX[i][9] = Ksets[0][i].homegoals[0];
    trainingX[i][10] = Ksets[0][i].homegoals[1];
    trainingX[i][11] = Ksets[0][i].awaygoals[0];
    trainingX[i][12] = Ksets[0][i].awaygoals[1];
    trainingX[i][13] = Ksets[0][i].weekendmatch;
    trainingX[i][14] = Ksets[0][i].cornerhome[0];
    trainingX[i][15] = Ksets[0][i].cornerhome[1];
    trainingX[i][16] = Ksets[0][i].corneraway[0];
    trainingX[i][17] = Ksets[0][i].corneraway[1];
    trainingX[i][18] = Ksets[0][i].freekickhome[0];
    trainingX[i][19] = Ksets[0][i].freekickhome[1];
    trainingX[i][20] = Ksets[0][i].freekickaway[0];
    trainingX[i][21] = Ksets[0][i].freekickaway[1];
    trainingY[i] = Ksets[0][i].y;
}

vector<vector<double>> validationX(Ksize,vector<double>(22));
vector<int> validationY(Ksize);
for (int i = 0; i < Ksize; i++){
validationX[i][0] = Ksets[1][i].elodif;
validationX[i][1] = Ksets[1][i].elodif2;
validationX[i][2] = Ksets[1][i].elomean;
validationX[i][3] = Ksets[1][i].elomean2;
validationX[i][4] = Ksets[1][i].distlog;
validationX[i][5] = Ksets[1][i].lessthan15km;
validationX[i][6] = Ksets[1][i].importance[0];
validationX[i][7] = Ksets[1][i].importance[1];
validationX[i][8] = Ksets[1][i].importance[2];
validationX[i][9] = Ksets[1][i].homegoals[0];
validationX[i][10] = Ksets[1][i].homegoals[1];
validationX[i][11] = Ksets[1][i].awaygoals[0];
validationX[i][12] = Ksets[1][i].awaygoals[1];
validationX[i][13] = Ksets[1][i].weekendmatch;
validationX[i][14] = Ksets[1][i].cornerhome[0];
validationX[i][15] = Ksets[1][i].cornerhome[1];
validationX[i][16] = Ksets[1][i].corneraway[0];
validationX[i][17] = Ksets[1][i].corneraway[1];
validationX[i][18] = Ksets[1][i].freekickhome[0];
validationX[i][19] = Ksets[1][i].freekickhome[1];
validationX[i][20] = Ksets[1][i].freekickaway[0];
```

```
    validationX[i][21] = Ksets[1][i].freekickaway[1];
    validationY[i] = Ksets[1][i].y;
}

    OLR statistics;

    statistics.estimateParameters(trainingX, trainingY);

    vector<double> beta, theta;


    statistics.getParameters(beta, theta);

    for (int i = 0; i < beta.size(); i++) {
        cout << beta[i] << "\t";
    }
    cout << endl;

    for (int i = 0; i < theta.size(); i++) {
        cout << theta[i] << "\t";
    }
    cout << endl;


    for (int i = 0; i < Ksize; i++) {
      statistics.calculateProbabilities(Ksets[1][i].probs, validationX[i], beta, theta);
    }



    float L_threshold = 0.0;
    float threshold = 1.0;
    int bet;
    int bets = 0;
    int unit = 1;
    double income = 0;
    for (int i = 0; i < Ksize; i++) {
        bet = 0;
        for (int j = 1; j < 3; j++) {
            if (Ksets[1][i].odds[j]*Ksets[1][i].probs[j] > Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet]) {
                bet = j;
            }
        }
        if ((1/Ksets[1][i].odds[bet]) > L_threshold && Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet] > threshold) {
            bets += 1;
```

```
            if (bet == Ksets[1][i].y-1) {
                income += unit*Ksets[1][i].odds[bet];
            }
        }
    }
    double profit = income-unit*bets;

    cout << "Your profit was " << profit << " units." << endl;

    cout <<"You played on " <<bets<< " games."<< endl;


    return 0;

    }
```

Code for initial run with parameters obtained in by compass search (simulation and search):

```
//
//  main.cpp
//  Master Thesis
//
//  Created by Hans Jacob Brun on 25/01/2019.
//  Copyright © 2019 Hans Jacob Brun. All rights reserved.
//
```

//Simulation for training and validation set

```
float simulating(match** Ksets, bool training, float threshold, float L_threshold){
    int bet;
    int bets = 0;
    int unit = 1;
    double income = 0;
    if (training) {
        for (int i = 0; i < datasize-Ksize; i++) {
            bet = 0;
            for (int j = 1; j < 3; j++) {
                if (Ksets[0][i].odds[j]*Ksets[0][i].probs[j] > Ksets[0][i].odds[bet]*Ksets[0][i].probs[bet]) {
                    bet = j;
                }
            }
```

```
            if ((1/Ksets[0][i].odds[bet]) > L_threshold && Ksets[0][i].odds[bet]*Ksets[0][i].probs[bet] > threshold) {
                bets += 1;
                if (bet == Ksets[0][i].y-1) {
                    income += unit*Ksets[0][i].odds[bet];
                }
            }
        }
    } else {
        for (int i = 0; i < Ksize; i++) {
            bet = 0;
            for (int j = 1; j < 3; j++) {
                if (Ksets[1][i].odds[j]*Ksets[1][i].probs[j] > Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet]) {
                    bet = j;
                }
            }
            if ((1/Ksets[1][i].odds[bet]) > L_threshold && Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet] > threshold) {
                bets += 1;
                if (bet == Ksets[1][i].y-1) {
                    income += unit*Ksets[1][i].odds[bet];
                }
            }
        }
    }
    double profit = income-unit*bets;
    BETS = bets;
    return profit;
}
```

//Limit-value search

```
float* local_opt_for_threshold(match** Ksets){
    bool flag = true;
    float best_threshold = 1.0;
    float best_l_threshold = 0.0;
    float max_profit = simulating(Ksets, true, best_threshold, best_l_threshold);
    //cout<<max_profit<<endl;
    float *thr = new float[4];
    float *l_thr = new float[4];
    thr[0] = 0.05;
    thr[1] = 0.0;
    thr[2] = -0.05;
    thr[3] = 0.0;
    l_thr[0] = 0.0;
    l_thr[1] = -0.05;
```

```
  l_thr[2] = 0.0;
  l_thr[3] = 0.05;
  while(flag){
     flag = false;
     for(int i = 0; i < 4;  i++){
        float temp_thr = best_threshold;
        float temp_l_thr = best_l_threshold;
        float profit;
        if(temp_thr+thr[i] >= 0 && temp_l_thr+l_thr[i] >= 0){
           temp_thr+=thr[i];
           temp_l_thr+=l_thr[i];
           profit = simulating(Ksets, true, temp_thr, temp_l_thr);
           //printf("Threshold : %f, L_threshold : %f, profit : %f\n", temp_thr, temp_l_thr, profit);
           if(profit>max_profit){
              flag = true;
              max_profit = profit;
              best_l_threshold = temp_l_thr;
              best_threshold = temp_thr;
           }
        }
     }
  }
  printf(" Best threshold : %f, Best L_threshold : %f, Best profit : %f", best_threshold, best_l_threshold,
max_profit);
  float* answer = new float[3];
  answer[0] = max_profit;
  answer[1] = best_threshold;
  answer[2] = best_l_threshold;
  return answer;
}
```

```
int main () {
```

//Search for tuning parameters

```
     for (int i = 0; i < datasize-Ksize; i++) {
        statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta, theta);
     }
```

```
float* best_tripple = local_opt_for_threshold(Ksets);
float max_profit = best_tripple[0];
float best_threshold = best_tripple[1];
float best_l_threshold = best_tripple[2];

int best_bets = BETS;

bool flag = true;
vector<double> beta_temp, theta_temp;

theta_temp = theta;
beta_temp = beta;

int count = 0;
while(flag){
    flag = false;
    printf("\nNew loop %d\n", ++count);
    for(int i = 1; i < theta.size()-1; i++){
        double delta_theta = theta_error[i];
        for(int j = 0; j < 2; j++){
            printf("\n %d %d\n",i,j);
            theta_temp[i] = theta[i] + delta_theta;
            //simulate
            for (int i = 0; i < datasize-Ksize; i++) {
                statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
            }
            float* answer = local_opt_for_threshold(Ksets);
            float temp_profit = answer[0];
            if(temp_profit > max_profit){
                theta[i] = theta_temp[i];
                max_profit = temp_profit;
                best_bets = BETS;
                best_threshold = answer[1];
                best_l_threshold= answer[2];
                flag = true;
                break;
            }
            delta_theta *= -1;
        }
        theta_temp[i] = theta[i];
    }
    for(int i = 0; i < beta.size(); i++){
        double delta_beta = beta_error[i];
```

```
            for(int j = 0; j < 2; j++){
                printf("\n %d %d\n",i,j);
                beta_temp[i] = beta[i] + delta_beta;
                //simulate
                for (int i = 0; i < datasize-Ksize; i++) {
                    statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
                }
                float* answer = local_opt_for_threshold(Ksets);
                float temp_profit = answer[0];
                //end of simulation
                if(temp_profit > max_profit){
                    beta[i] = beta_temp[i];
                    max_profit = temp_profit;
                    best_bets = BETS;
                    best_threshold =answer[1];
                    best_l_threshold= answer[2];
                    flag = true;
                    break;
                }
                delta_beta*= -1;
            }
            beta_temp[i] = beta[i];
        }
    }

    printf("\nBest of the bests is - profit: %f, threshold: %f, l_threshold: %f\n", max_profit, best_threshold,
best_l_threshold);
    for (int i = 0; i < beta.size(); i++) {
        cout << beta[i] << "\t";
    }
    cout << endl;

    for (int i = 0; i < theta.size(); i++) {
        cout << theta[i] << "\t";
    }
    cout << endl;


     for (int i = 0; i < Ksize; i++) {
     statistics.calculateProbabilities(Ksets[1][i].probs, validationX[i], beta, theta);
     }

    float validation_profit = simulating(Ksets, false, best_threshold, best_l_threshold);

    int validation_bets = BETS;
```

```
    printToFile(fout, k, s++, validation_bets, validation_profit, best_threshold, best_l_threshold);
}


fout.close();
return 0;
}
```

//Search for reduction in step-length when no further improvements can de done

```
    bool flag = true;
    vector<double> beta_temp, theta_temp;

    theta_temp = theta;
    beta_temp = beta;

    int count = 0;
    while(flag){
        flag = false;
        printf("\nNew loop %d\n", ++count);
        for(int i = 1; i < theta.size()-1; i++){
            double delta_theta = theta_error[i];
            for(int j = 0; j < 2; j++){
                printf("\n %d %d\n",i,j);
                theta_temp[i] = theta[i] + delta_theta;
                //simulate
                for (int i = 0; i < datasize-Ksize; i++) {
                    statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
                }
                float* answer = local_opt_for_threshold(Ksets);
                float temp_profit = answer[0];
                //end of simulation
                if(temp_profit > max_profit){
                    theta[i] = theta_temp[i];
                    max_profit = temp_profit;
                    best_bets = BETS;
                    best_threshold = answer[1];
                    best_l_threshold= answer[2];
                    flag = true;
                    break;
```

```
            }
            delta_theta *= -1;
        }
        theta_temp[i] = theta[i];
    }
    for(int i = 0; i < beta.size(); i++){
        double delta_beta = beta_error[i];
        for(int j = 0; j < 2; j++){
            printf("\n %d %d\n",i,j);
            beta_temp[i] = beta[i] + delta_beta;
            //simulate
            for (int i = 0; i < datasize-Ksize; i++) {
                statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
            }
            float* answer = local_opt_for_threshold(Ksets);
            float temp_profit = answer[0];
            //end of simulation
            if(temp_profit > max_profit){
                beta[i] = beta_temp[i];
                max_profit = temp_profit;
                best_bets = BETS;
                best_threshold =answer[1];
                best_l_threshold= answer[2];
                flag = true;
                break;
            }
            delta_beta*= -1;
        }
        beta_temp[i] = beta[i];
    }

    for(int i = 1; i < theta.size()-1; i++){
        double delta_theta = theta_error[i] / 2;
        for(int j = 0; j < 2; j++){
            printf("\n %d %d\n",i,j);
            theta_temp[i] = theta[i] + delta_theta;
            //simulate
            for (int i = 0; i < datasize-Ksize; i++) {
                statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
            }
            float* answer = local_opt_for_threshold(Ksets);
            float temp_profit = answer[0];

            if(temp_profit > max_profit){
                theta[i] = theta_temp[i];
```

```
                max_profit = temp_profit;
                best_bets = BETS;
                best_threshold = answer[1];
                best_l_threshold= answer[2];
                flag = true;
                break;
            }
            delta_theta *= -1;
        }
        theta_temp[i] = theta[i];
    }
    for(int i = 0; i < beta.size(); i++){
        double delta_beta = beta_error[i] / 2;
        for(int j = 0; j < 2; j++){
            printf("\n %d %d\n",i,j);
            beta_temp[i] = beta[i] + delta_beta;
            //simulate
            for (int i = 0; i < datasize-Ksize; i++) {
                statistics.calculateProbabilities(Ksets[0][i].probs, trainingX[i], beta_temp, theta_temp);
            }
            float* answer = local_opt_for_threshold(Ksets);
            float temp_profit = answer[0];
            //end of simulation
            if(temp_profit > max_profit){
                beta[i] = beta_temp[i];
                max_profit = temp_profit;
                best_bets = BETS;
                best_threshold =answer[1];
                best_l_threshold= answer[2];
                flag = true;
                break;
            }
            delta_beta*= -1;
        }
        beta_temp[i] = beta[i];
    }

}
```

//Simulation with removal of matches containing Manchester United or Liverpool from validation fold.

```
    } else {
        for (int i = 0; i < Ksize; i++) {
            bet = 0;
            for (int j = 1; j < 3; j++) {
                if (Ksets[1][i].odds[j]*Ksets[1][i].probs[j] > Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet]) {
                    bet = j;
                }

            }
            if ((1/Ksets[1][i].odds[bet]) > L_threshold && Ksets[1][i].odds[bet]*Ksets[1][i].probs[bet] > threshold) {
                if (Ksets[1][i].home != "MAN UNITED" && Ksets[1][i].away != "MAN UNITED" && Ksets[1][i].home !=
"LIVERPOOL" && Ksets[1][i].away != "LIVERPOOL")
                    bets += 1;
                if (bet == Ksets[1][i].y-1) {
                    income += unit*Ksets[1][i].odds[bet];
                }
            }
        }
    }
```