MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Bula Radek** Personal ID number: **484298**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Computer Science**

Study program: **Open Informatics**

Specialisation: **Artificial Intelligence**

II. Master's thesis details

Master's thesis title in English:

Strategies for pre-match trading on sports betting exchanges

Master's thesis title in Czech:

Strategie pro předzápasové sázky na sportovních burzách

Guidelines:

The rise of betting exchanges enabled new possibilities of sports betting. Together with all the available data sources precisely mapping the sports statistics, it naturally leads to the idea of an automated betting system. One of the specifics of betting exchanges is the movement of pre-match odds. The goal of this thesis is to develop, evaluate and optimize strategies, which attempt to profit in this pre-match period.
1. Research the domain of pre-match odds movement.
2. Select a specific sport. Prepare and process suitable historical data.
3. Design multiple betting strategies and optimize them with a focus on increasing long-term profit.
4. Use appropriate machine learning algorithm for the prediction part of the strategy.
5. Evaluate and compare designed strategies using suitable metrics.

Bibliography / sources:

[1] Dzalbs, Ivars, and Tatiana Kalganova. "Forecasting Price Movements in Betting Exchanges Using Cartesian Genetic Programming and ANN." Big data research 14 (2018): 112-120.
[2] Gonçalves, Rui, et al. "Deep Learning in Exchange Markets." Information Economics and Policy (2019).
[3] Bebbington, Peter Antony. Studies in informational price formation, prediction markets, and trading. Diss. UCL (University College London), 2017.
[4] Bunyan, Andrew. Time Series Analysis and Forecasting of In-Play Odds on a Betting Exchange. Diss. Dublin, National College of Ireland, 2015.

Name and workplace of master's thesis supervisor:

Ing. Matej Uhrín, Department of Computer Science, FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **11.02.2020** Deadline for master's thesis submission: **05.01.2021**

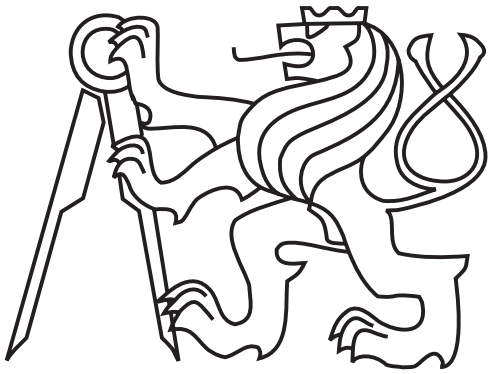
Assignment valid until: **30.09.2021**

Ing. Matej Uhrín Supervisor's signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature



Master's thesis

Strategies for pre-match trading on sports betting exchanges

Bc. Radek Bula

Department of Computer Science
Supervisor: Ing. Matej Uhrín

December 30, 2020

Acknowledgements

I would like to thank my supervisor Ing. Matej Uhrín for the opportunity to work on an interesting topic, for his valuable advice and guidance. Second, I would like to thank the faculty for creating an environment for writing this work. And finally, I wish to thank all the precious people close to me, especially my family.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on December 30, 2020 .....

Czech Technical University in Prague
Faculty of Electrical Engineering

Citation of this thesis

Bula, Radek. *Strategies for pre-match trading on sports betting exchanges.* Master's thesis. Czech Technical University in Prague, Faculty of Electrical Engineering, 2020.

Abstrakt

Cílem této práce bylo prozkoumat oblast vývoje předzápasových kurzů na sportovních burzách. Práce se zaměřuje na popis technikálií a nástrojů používaných k sázení na sportovní události. Na základě provedené rešerše je vybrán konkrétní sport (dostihy) a vhodná data. Pro účely následné analýzy je provedeno předzpracování, čištění a formátování těchto dat. Prováděná analýza popisuje základní vlastnosti trhu a jejich vývoj ve sledovaném čase. Se zohledněním výsledků analýzy je navrženo několik základních strategií, které využívají algoritmy strojového učení. Každá ze strategií je parametrizována tak aby při jejich následné evaluaci došlo k otestování co nejvíce možností. Na základě evaluace podle definovaných metrik je vybráno několik strategií, které budou dále vylepšovány. Vylepšování spočívá ve využívání umělých neuronových sítí různých architektur pro predikční část strategie. Nakonec jsou popsány a porovnány nejlepší strategie a jejich výsledky.

Klíčová slova sportovní sázení, předzápasové sázení, sázkové burzy, strategie, dostihy

Abstract

The goal of this thesis is to explore a field of pre-match price development on sports exchanges. The thesis presents a description of the techniques and tools used in betting on sports events. According to past research is chosen specific sport (horse racing) and appropriate data. For purposes of the following work is applied data preprocessing, cleaning, and reshaping. A performed analysis then describes the basic properties of a market and their development in observed time. With respect to this analysis are proposed several strategies, which use machine learning algorithms. Each from strategies is parametrized in order to increase the number of tested and evaluated scenarios. Based on the evaluation and according to defined metrics is selected a set of strategies intended for further improvement. This improvement lies in using artificial neural networks of different architectures for the prediction part of the strategy. In the end, the best strategies are described, compared, and commented.

Keywords sports betting, pre-match betting, betting exchange, strategies, horse racing

Contents

List of Figures

List of Tables

Introduction

Sports betting has a long tradition in the UK, as the first bets were placed centuries ago. The popularity of gambling is closely linked to the offer of events for which the outcome is guessed. Horse racing has been accompanied by betting for the longest time from all known sports and even in the current age of football and other ball games, horse racing still keeps its popularity in the UK, where the average number of major races per day exceeds 30. [1]

The current century and its digital technologies made it possible to place a bet from the comfort of home within few seconds. Betting thus becomes more accessible and popular. In 2000, the first betting exchange (BetFair) was created - a platform that allows betting between individual punters, while in the classic model, the bet is placed between bookmaker and gambler. Thanks to these much more favorable conditions for classic gambling (due to lower commission), the possibility of trading has arisen. [2]

The gambler guesses the outcome of a certain event, he accepts the odds and either wins or loses. The trader on the other hand tries to estimate the direction of the odds development and make money on the change. As reality develops, the trader can adjust his trade, increase the amount, exit early, or keep waiting. The trading scenario offers many possibilities compared to the gambler's win/loss outcome. Betting traders are similar to financial traders. They also buy and sell to make a profit. Only the subject of the trade is not a stock or other commodity but the odds of the event outcome.

We have two basic ingredients - repeating events such as horse racing and a platform for trading. The idea of exploiting the market's behavior easily comes up, maybe some repeating patterns or mechanisms are included in the markets. The knowledge of them could be turned into profit, but obtaining such knowledge can be time and computationally expensive. And as people are lazy, they like to get their problem solved by someone else, in this case by the computer.

Machine learning (ML) is our third ingredient. ML is a group of algorithms that builds and learn a mathematical model based on sample data, known as the "training data", which describe the underlying problem. The model then makes predictions without being explicitly programmed to do so. Each algorithm has many hyper-parameters, which are defining and controlling the learning process. At the end of the day, the main requirement is to build and learn a model, which predicts outputs as good as possible on "unseen data" given the objective function and "training data". We have to define our problem and objective to some of these algorithms and ensure the process of learning. [3]

This work is divided into six chapters and a conclusion. We will start in chapter 1 by describing the principles of the betting world, their participants, and possibilities of profit. The next chapter 2 will contain the summary of so far provided research on similar topics. Then, in chapter 3, the input data, their formatting and reshaping will be described. Chapter 4 is exploring market properties and extracting some conclusions about their features. In the following chapters, 5, 6, several strategies are designed. Additionally, the definition of the strategy itself and metrics to compare them are provided. Finally, the conclusion summarizes all the chapters of the thesis, evaluates the results, and proposes ideas for further work.

Chapter 1

Betting world

In this chapter, we will describe several betting concepts and principles. Two main betting platforms - classic bookmaker and betting exchange will be briefly introduced.

1.1 Sports betting

Sports betting is a game restricted by predefined rules with two participants: in a typical scenario bookmaker (bookie) and gambler (punter). Bookmaker provides an offer of bets with defined odds. Gambler observes this offer and according to his strategy (see 1.1.2) can place a bet. After the event is realized, the bet is evaluated. If the gambler was right, he wins, otherwise he loses.

The subject of bet can be event outcome (winner), goal scorer, number of goals, number of football corners, final score, time spans between goals, etc. The diversity of bet subjects depends on the type of sport and the interest of gamblers. This work specifically aims at the event outcome bets.

1.1.1 Odds formats

There are multiple formats of odds around the world. They can easily be converted to one another as they all describe the same property - probability. The odds are sometimes referred to as the price. [4]

- **Probability** - each odds format is related to probability which reflects the chance of a specified outcome to happen. In a fair environment, the sum of all possible outcomes should be 100 %.
- **Decimal odds** - are used mostly in Europe. Odds are represented in decimal numbers, the potential payout can be easily calculated as the odds multiplied by the bet. Profit is then calculated as payout reduced by the original bet.

For example, if 100 CZK bet is placed at odds 4.0, potential payout and profit is 400 CZK, respectively 300 CZK.

- **Fraction odds** - originally comes from the UK. Odds are represented by a fraction, a numerator represents profit, which can be made if the bet is placed at a denominator value.
  For example, if 100 CZK bet is placed at odds $\frac{9}{1}$, potential payout and profit is 1000 CZK, respectively 900 CZK.

- **American odds** - used mostly in the USA, odds can be negative or positive. A positive number represents the amount of money that can be won if a bet with 100 value of money was placed. A negative number represents the absolute value of money that has to be placed to win an amount of 100.
  For example, if 100 CZK bet is placed at odds -200, potential payout and profit is 150 CZK, respectively 50 CZK. Or if 100 CZK bet is placed at odds 200, potential payout and profit is 300 CZK, respectively 200 CZK.

1.1.2 Betting strategy

There can be several strategies for betting. Some gamblers are wagering on their favorite team or player. Others are looking for low odds, believing that the probability of losing is lower. There are also gamblers who use statistics, their decisions are then more rational than emotional.

Most of the gamblers are comparing the odds given by the bookmaker to their own estimation about the event result. If there is a difference between given odds and an estimate (based on emotions, insider info, or some evidence), the gambler has a reason to place a bet.

1.2 Bookmaker

Bookmaker is an organizer of bet, he ensures offer of bets, provides multiple sale channels (web, phone, street branch). He keeps the information about gambler account, withdrawals, deposits, betting history, etc. Bookmaker also has very detailed statistics and complex models that are, together with an expert team, used for estimating odds. Estimated odds are then "adjusted" in the bookmaker's favor, that is how the bookmaker makes his profit. [4]

1.2.1 How bookmaker makes a profit

Events such as a football match (including overtime) or tossing a coin have only two possible outcomes. If the coin is fair, we can say, that the probability of both outcomes is equal - 50 %.

In the coin case, the probability table with potential win/loss looks like this.

	Head	Tail	Margin
Odds	2.0	2.0	
Implied probability	50 %	50 %	0 %
Gambler win/loss	+1 / −1	+1 / −1	

Table 1.1: Fair betting environment.

Then the adjusting comes in, the bookmaker adds some margin and spread it between event outcomes. In this case, the margin is 7.5 %.

	Head	Tail	Margin
Odds	1.85	1.85	
Implied probability	53.75 %	53.75 %	7.5 %
Gambler win/loss	+0.85 / −1	+0.85 / −1	

Table 1.2: Betting environment modified by bookmaker's margin.

$$\text{margin} = \frac{\sum_{j=1}^{K} \frac{1}{o_j} - 1}{\sum_{j=1}^{K} \frac{1}{o_j}} \tag{1.1}$$

Result of the equation 1.1 is called the bookmaker's margin[1], and represents the negative expected value of the game given the odds.

The next experiment (figure 1.1) provides a comparison between betting with and without margin. The player in both cases places bets only on flipping a tail. As the coin gets actually flipped, player's bankroll is increased or decreased by the amount specified in tables 1.1 (blue color) and 1.2 (orange color). Both simulations use the same sequence of flipped sides. In the 0 % margin case, the lucky player ended with 5 % profit. In the 7.5 % margin case, the player ended up with 14 % loss. The fact is caused by the difference in the size of win and loss in the 1.2 table.

[1]Often wrongly calculated as simply the remainder over 1.0 as $\sum_{j=1}^{K} \frac{1}{o_j} - 1$

Figure 1.1: Comparison of long term margin impact on bankroll.

Bookmaker's odds adjusting technique is quite obvious on a simple example like this one, but a similar methodology is applied to all of the offered bets. Precise estimation of event outcome with the addition of profit margin puts bookie in a statistically better position than the gambler has.

1.2.2 Back side, lay side

Two possible types of bet can be placed on a certain outcome, a lay side (bookmaker) makes a profit when the outcome is different from the placed bet. A back side (gambler) makes a profit if the outcome is the same as in placed bet. In other words, the lay side is saying, that specified outcome **does not** happen, while the back side is saying that specified outcome **does** happen.

Assume a football match between Slavia and Barcelona, the decimal odds on the draw outcome is 4.5. A gambler places a 100 CZK bet on the draw. The following table specifies potential net profit and loss for both roles.

	Draw	Different outcome
Lay side (bookmaker)	−350 CZK	+100 CZK
Back side (gambler)	+350 CZK	−100 CZK

Table 1.3: Example of back and lay side net profit and loss.

1.3 Betting exchange

Betting exchanges unlike classic bookmakers provide bets where no odds are specified. Exchange only ensures offer in sense of bet type, possible outcomes, evaluation rules, etc. The whole platform lays on punters, which are making and taking offers (odds), they can place both types of bet – back and lay. In

other words, punters are not placing a bet but making an offer.
The exchange algorithm is then connecting back and lay offers with identical odds on a certain outcome. Two offers are after connecting turned into usual placed bets (offers get matched). This whole process of connecting and matching happens in a single exchange loop. Exchange in regular traffic without overload make several loops per second.

The punter can see only aggregated offers – side, odds, available amount. No detailed information about a single offer is provided. If the punter in this aggregated list spots an interesting opportunity, he can easily make the opposite offer which should be matched within a few next loops. He can also make an offer that at the current time does not have the obvious opposite offer to be matched with, the offer can be matched later or does not get matched at all.

These odds are truly reflecting the overall estimation over all punters, no expert knowledge or margin adjusting is applied. Exchange profit is made as a percentage charge from bets which yield to profit, usually between 1 % and 3 %. [5]

1.3.1 Odds change

As new information about the event enters the public, estimation of the outcome starts changing. If Leo Messi breaks his leg, Barcelona's chance to win over Slavia gets slightly lower. These facts are reflected in the odds changing.

The bookie has to correct his odds manually, to prevent bets placed on obviously incorrect odds, being fast is crucial. Exchanges thanks to their autonomy have a much favorable position, punters easily update their offers, the holder of potential loss is always the punter, not the exchange.

1.4 Profit spots

The betting world offers several opportunities to make a profit. The most popular of them are listed in this section.

1.4.1 Matched betting

Matched betting is a technique that guarantees a profit from the promotions offered by bookmakers. Betting exchanges typically do not offer promotions (except the welcome bonus). But almost all bookies advertise those offers to attract new customers or remind their existence to current customers. The advantage of this betting opportunity is that those offers are just waiting to be taken. No extra knowledge or skill is required. The net profit per promotion is in the order of hundreds of CZK. Disadvantage is the limited number of these offers furthermore, bookie often detects and blocks players, which are betting

only in sense of exploiting promotions. To make matched betting long-term business, regular bets have to be placed too. [6]

1.4.2 Exchange trading

Odds moving with a combination of exchange possibilities (back and lay side) yields to a specific opportunity to make a profit. If a certain outcome is selected and both back and lay bets are placed with a suitable difference in odds, guaranteed profit can be made. In other words, knowledge of price movement can be monetized. The principle is similar to currency markets, for example, the USD/EUR exchange. [5]

There are two basic ways how to trade in the sports betting exchange market.

- **Scalping** - is a trading technique that tries to make a profit on even little price movements. This technique is mostly used in more stable markets.
- **Swinging** - swing trading tries to spot bigger price movements. This type of trading is more suited to volatile markets where the price drifts on larger scales.

The advantage of this method is that trading opportunities are available almost every day. The problem can be that for long-term profit some insider info or trading skill is required.

1.4.3 Arbitrage betting

Arbitrage is a risk-free method of betting, which is taking advantage of inequalities between bookmakers and/or exchanges. To make an arbitrage bet one specific event has to be chosen. As the event develops, odds are slightly changing. This is the moment when multiple organizers of bet can have different odds. If the difference between them makes an opportunity for placing a bet on all possible outcomes with guaranteed profit without dependence on the result, the combination of placed bets is called arbitrage. Many arbitrage opportunities exist for only a very short interval. In most cases, all bets have to be placed in a few seconds, otherwise, the opportunity disappears. Spotting an arbitrage opportunity can be complicated as the number of included bookies and exchanges has to be high, also the data have to be as actual as possible. [7]

Chapter 2

Research

Most existing papers in the field of sports betting focuses on outcome prediction. Odds movement in pre-race time is a very rare subject of research. With that being said, some relevant work in this area has already been done.

2.1 Interesting works

In [8] author implements a short-term forecasting system where various neural network architectures are used for prediction (DNNC, LSTM, CNN). The trained and tested dataset consists of horse racing win markets in the time period from 2014 to 2016. The paper is focused only on the last 10 minutes of pre-race time because the highest market activity was inspected in this period. To identify different types of market dynamics, a rule-based decision tree with 647 categories were used. In each category, where the minimum race count criterion was met, three types of neural networks were trained and tested. Data used in these networks were only market data processed into several indicators. Observations were generated using the sliding window technique with variable input density. The output corresponds to the price change between time t and time $t + 90$, price change is according to its size converted into one of five classes. Each class represents an interval of price change (strong up, weak up, neutral, weak down, strong down). The final prediction task is a classification problem where softmax is used as an output activation function and categorical cross-entropy is used as a loss function.
The paper concludes that the CNN network outperforms more suitable LSTM networks. However, the overall profit after more than thousands of trades is only 1.35 GBP. Considering that each trade opening bet value is 3 GBP, the overall profit is negligible. Additionally, the paper suggests categorization of markets into groups according to different observed properties.

In [9], the authors explore ways to forecast price movements using trading strategies based on Artificial Neuron Networks (ANN) and Cartesian Genetic

Programming (CGP). Dataset was collected from January to May of 2016 and again consists of horse racing win markets. The paper also proposes that markets are most active in the last 10 minutes before the start time. The prediction output corresponds to a price change between two times. In this case: between 6 and 0 minutes (before the race starts). Each horse is used as a single observation. Several strategies (input features) and several objective metrics (what is optimized) were introduced in this work. Authors do not compare them directly, but their performance differs.
From total of 36 strategies (input features and objective combinations), 8 were profitable from which only 2 were significantly profitable. The best strategy had 2.95% efficiency. Work does not provide direct conclusions on used strategies. However, both profitable strategies were using profit objective. As the profit objective function directly reflects the underlying problem - maximizing profit, using this objective can be very beneficial compared to common methods such as MSE (Mean Square Error).

The paper [10] is, compared to the previously mentioned papers, relatively old. Moreover, unlike them, it is focused on in-play betting strategies for tennis markets. In-play betting is more volatile and unstable compared to the pre-race. But tennis matches have gradual development, winning a game, set or match typically does not happen instantly. In football, there are penalty kicks or unexpected goals, in horse racing unexpected fails, or pull-ups. The author does not provide info about the dataset collection period, inspected markets are tennis win markets (always only two possible outcomes). Work proposes a custom loss function designed for purposes of betting exchange. Designed function focuses more on profitability than on the distance between predicted and target value. The sliding window technique is used again, window size is fixed at 30 seconds.
In all of the tested scenarios, the custom loss function outperformed the standard loss function. However, the author also notes that the proposed loss function may not be ideal for the general odds trading tasks, finding the true optimal loss function will require detailed analysis.

2.2 Research conclusions

Some interesting concepts were described: such as the sliding windows technique or custom loss function focused directly on profit optimization. Works also link other useful sources, but they are not primarily focused on the same problem as this thesis.

Chapter 3

Data pre-processing

3.1 Picking a sport field

The field of pre-race betting is relatively new and most of the works are focused on horse racing. They provide some findings and interesting ideas to follow, as have been mentioned in the previous chapter. The typical horse race has a characteristic pre-race ceremony - the horses (runners) are together with jockeys introduced to the audience and then brought to the course and placed into gates. During this time a major amount of bets is placed, the market gets more activity, runner prices are changing and opportunities are arising.

This support of existing resources and pre-match ceremony uniqueness is why we chose horse racing as the main sport-field of this thesis.

3.2 Original data source

The data are originally stored in MySQL relational database. More than 50 tables are storing over 30 GB of data. Data collection was performed by PHP server-side app, which was scraping the data sources - sports exchange and web portal providing summary information on upcoming races (racecards) with the historical horse statistics. PHP application also provides a web front-end interface for user-friendly data browsing and visualizing. Realization of this described workflow was not part of this thesis, but the provided data were served as input for this thesis.

3.3 Data description

In this section, we will describe the structure of data that were collected and processed into training, validation, and testing datasets.
The "event" in figure 3.1 means the physically happening competition. The race and the market are describing two different perspectives on the event.

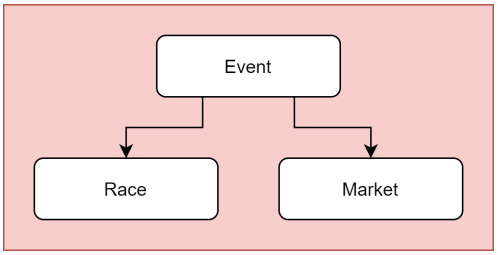


Figure 3.1: Composition of data source.

- **Race** - includes the data from the racecards portal. Data describes the race, its conditions, and individual runners. Data are given several hours before the race starts and they do not change in time.
- **Market** - includes the data from sports exchange. Data describes the market state and aggregated offers for individual runners. Data are collected every 2 second in the time window from 30 minutes before the race starts until the race finishes. That is approximately 1200 market states per single event.

3.3.1 Race data

Race data describe the unchanging properties of the event. The main entity is the horse race itself.

- **Title** - public label of race.
- **Start date and time** - when the race should start.
- **Country** - in which country (UK/IRL) is the race taking place.
- **Course** - on what race-course is the race happening (about 50 possible courses across UK and IRL).
- **Winning** - the size of the race prize.
- **Type** - the basic characteristic of the race - flat race or jumps.
- **Class** - classification of the race quality (total 8 classes).
- **Going** - is the description of the ground, which is determined by the amount of moisture in the ground.
- **Distance** - specifies the length of the race track.
- **TV** - which television channels stream the race.
- **Runners** - list of the race participants, i.e. horses.

Anyway, the most important entities are individual horses. There are several properties each of them has:

- **Name** - name of the horse.
- **Age** - the actual age of the horse.
- **Last races** - list of the last 10 races (if they are available), with all the race data (described above), plus the final placing and winning/beaten distance.
- **Signs** - several signs indicating important results from the previous races.
  - BF - beaten favourite.
  - CW - recent win on the same course.
  - DW - recent win in the race with a similar distance.
- **Tips** - how many expert voters picked certain horse as the race favourite.
- **Comment** - comment of an expert about the expected horse performance in the race.
- **Rankings** - a triplet of rankings, which summarizes the actual horse form.
  - OFR - official rating.
  - TSR - top speed rating.
  - RPR - racing post rating.
- **Weight** - the amount of additional weight which the horse carries.
- **Crew** - the horse is always ridden by a **jockey**, trained by a **trainer** and owned by an **owner**. The whole crew influences the horse winning chance. Similar data are recorded for all of them:
  - **Last 14 days** - a ratio of participating to winning in races during the last 14 days.
  - **Seasonal data** - aggregated statistics from whole season specifying the number of races, win ratio, placing, earnings. All the data can be filtered by race type and country.
  - **Last races** - the same kind of data as for the horse.

3.3.2 Market data

The market data consists of the value of matched money and the moment when the race actually starts.

- **Matched money** - the total value of placed bets across all runners in the current market.
- **In play date and time** - the actual start of the race. The usual delay to regular start time is in the order of minutes and is not known in advance.

For individual runners, the market data consists of the following:

- **Back offers** - aggregated offer for the back side. Three best (highest) possible decimal back odds and the size of available bet amount for each of them. In the following pages, we refer to them as B1P, B1V, B2P, B2V, B3P, B3V. The B stands for back, P for price, V for value.
- **Lay offers** - aggregated offer for the lay side. Three best (lowest) possible decimal lay odds and the size of available bet amount for each of them. In the following text, we refer to them as L1P, L1V, L2P, L2V, L3P, L3V. The L stands for lay, P for price, V for value.
- **Voided** - the cancellation of the actual presence of the horse in the race. There are several reasons for this and it can happen until the race starts.

3.4 Data storage

It is crucial to access data-source as fast as possible in order to provide data analysis, generation of learning datasets, or other data-dependent operations. In this scope, the MySQL storage on a remote server is not an optimal solution. All the related data are divided and stored in various tables and accessing them over the internet and converting them into the required shape is redundant and time expensive. For transformation between MySQL and the local data storage solution we used the following workflow:

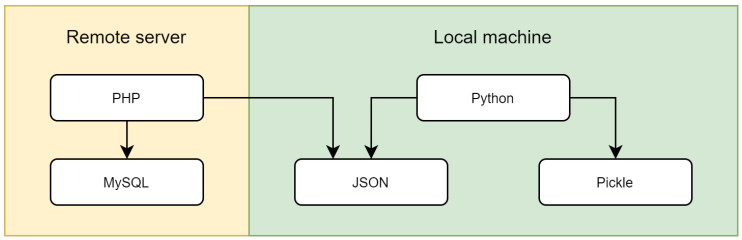


Figure 3.2: Transformation of data storage.

As the main programming environment for this thesis, we chose Python version 3.6 and the Pandas library was used for work with tabular data. Figure 3.2 describes the workflow of transformation between data sources.

In the first step, the PHP script accesses MySQL storage and queries all the relevant data for a single event. All that data is placed into data interchangeable format JSON and sent to local storage. This operation is performed for each of the circa 30 thousands of events in the whole dataset.
The second part is provided by Python, which sequentially loads each downloaded JSON file and processes data formatting and reshaping. Then the market and race data are stored using pickle serialization. Data stored in this way can be loaded about 70 % faster than using common JSON or CSV format. [11]

3.5 Data format

As have been mentioned, data is formatted and reshaped before it is stored in a pickle file. The formatting is understood as the entry-level of the data cleaning and validation process. The original raw data is converted into corresponding formats - *bool* , *int*, *float*, *string*. Date time formats are unified into timestamp values. Missing values are filled with *null*/*None*, as the originally stored zero value can be confusing in some cases. Some additional features are prepared, e.g. *to_start*, which represents the relative time in the market data. The last task of formatting is to detect huge anomalies in the data, e.g. completely missing exchange (market) data, or a different count of horses in race data and market data, detection of such anomalies sometimes leads to a removal of a whole event. All these actions are mandatory for data manipulation without errors. The data is processed and converted just once, the following actions can then assume that input data is valid and there is no need to waste time on a redundant verification.

By reshaping we mean placing all the relevant features together into a tabular shape. There are three kinds of tables used in horse racing case - race data, runner data, market data.

- **Race data** - in one event pickle file there is only a single row. A row represents race, columns represent race features.
- **Runner data** - in one event pickle file there are several rows. A row represents runner (horse), columns represent runner features.
- **Market data** - in one event pickle file there are several tables - one for each horse, in each table there are many rows. A row represents the time point, columns represent market data (back and lay offers, matched money, voided).

Chapter 4

Exploring market properties

4.1 Market dynamics

From the previously conducted research, it is obvious that the market has certain characteristics based on race parameters. Discovering the impact of the individual parameters could help us understand how the market works. Obtained knowledge could also be a good starting point for designing a successful betting strategy.

Profitable trade consists of two bets placed on both sides at different prices with sufficient value. There are several indicators used to describe the basic properties of the trading environment.

- **Price-change** - is an absolute difference between the price in time observation t and $t + 1$. A higher difference means higher opportunity to spot a profitable movement. Measured in decimal odds.
- **Back-lay value** - is the total value of money available for placing a bet, both on the back and the lay side. A higher value means that more people and offers are involved in the market. Measured in GBP.
- **Price-space** - describes the space between the highest back and lowest lay offer. Tight space ensures that even a small odds movement can lead to a profitable trade. Measured in decimal odds.

The above mentioned indicators were observed in the 30 minute pre-race interval. Observation density was 2 seconds, hence 900 values of each indicator have been collected for each runner. As the granularity of the collected data was too high, values of the single runner indicator from each minute were pushed into a single value describing the minute average. Finally, for each runner in the dataset, there are 90 (3 indicators $\times$ 30 minutes) values describing the evolution of specified indicators. Besides the global view i.e. how

markets work on average, individual (both race and runner) features can be examined.

4.1.1 Global overview

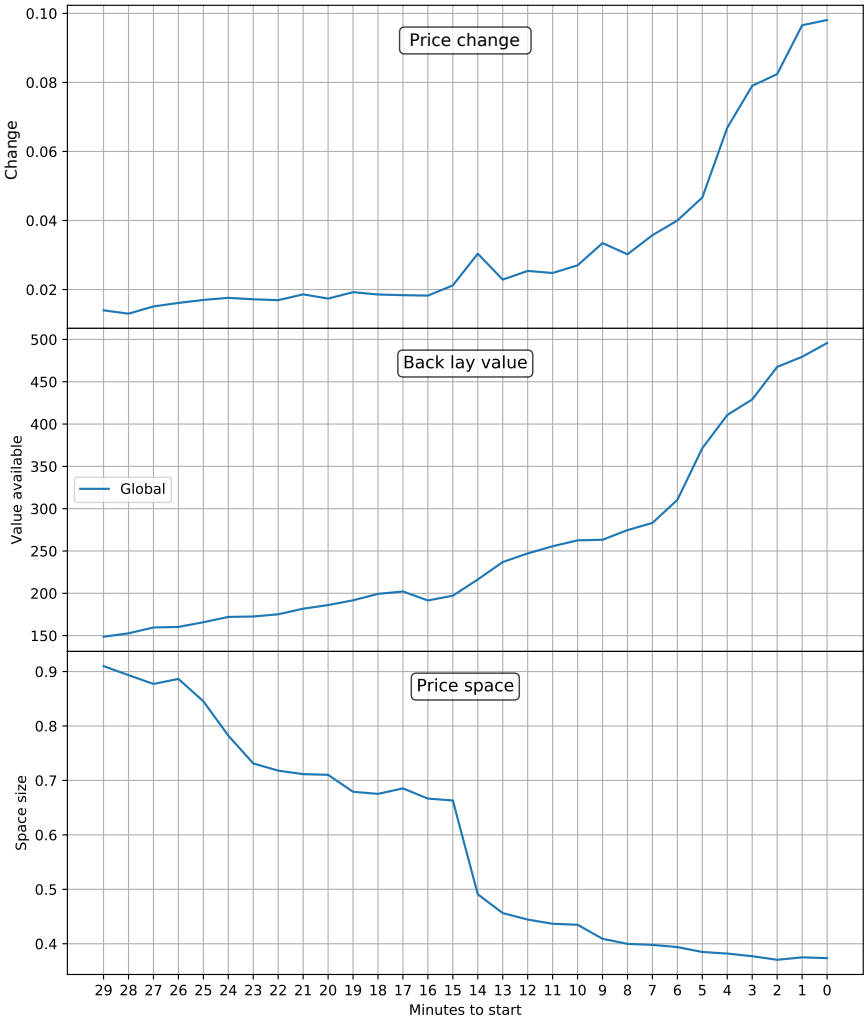


Figure 4.1: Evolution of basic market indicators in time, averaged over all runners in the dataset.

Figure 4.1 reveals basic trends for each indicator. With a decreasing number of minutes to start, price-change and back-lay value are rising, price-space is narrowing. These trends have an explanation, as the race/market gets more attention, more people interfere in trading. More people mean more bets and more bets improve market conditions. The overall value of the available bet offer gets larger, a higher amount of offer pushes down the price-space. And as people spot opportunities and place bets, overall price-change is growing.

Attention deserving is the moment between the 13th and 16th minute to start. Price-space suddenly gets significantly lower which is also reflected in the small peak in price-change at the same time. Just a short time before this moment the back-lay value growth slowed down with a tiny drop. Starting at this time, the markets are safe for trading, by safe we mean that the price-change and back-lay value are only growing and price-space is slowly decreasing.

4.1.2 Race and runner features

As have been mentioned, this analysis can also explore individual race or runner features.

- **Race class** - race class reflects the market quality. A total of 8 different classes are present in the dataset. The best class (i.e. class 1) has the biggest back-lay value, whereas price-space and price-change are the lowest. As the class gets higher (lower market quality), back-lay value gets lower, price-change, and space higher.
- **TV coverage** - the vast majority of races is covered by some TV channel, 6 mostly frequently channels are examined. The daily channels have very similar properties, but two, namely ITV and ITV4 differ. Back-lay value is significantly higher, price-change, and price-space is lower. Further investigation pointed out that these channels are broadcasting only popular/important races i.e. class 1, high winning prize, etc.
- **Winning** - is the money which belongs to the winner of the race. The discovered relation is similar to the first one. As the winning amount decreases (lower market quality), back-lay value gets lower, price-change, and price-space higher.
- **Race time** - different weekdays (7 days) and different hours (6 intervals) were studied. No visible relation was discovered, all indicators reported very similar results. Only one exception, starting time between 14:00 and 15:00 had a slightly higher back-lay value.
- **Runner price** - for each runner, we observed the price at 30 minutes before race start (opening price). Prices were then split into 6 intervals $\{[1-3), [3-6), [6-10), [10-15), [15-21), [21-1000]\}$, using decimal odds. Opening price can obviously move during the pre-race time, but the movement inside the size of the interval is negligible. Moreover, the occasional interval switch is not a big deal in case of a basic exploration of how the market properties are affected by the opening price.

  Categories with lower decimal odds (higher win probability) have higher back-lay value and price-change i.e. favorite runners tend to be the main subject of bets. The effect of price-space narrowing is observable mainly in lower odds, outsiders (higher odds) often keep their space.

- **Count of runners** - each race has a fixed count of participants - runners, this count also affects the properties of the market. Counts of runners were again split into several intervals. Races with fewer participants have higher price-change and back-lay value. Details can be found in the following figure 4.2.

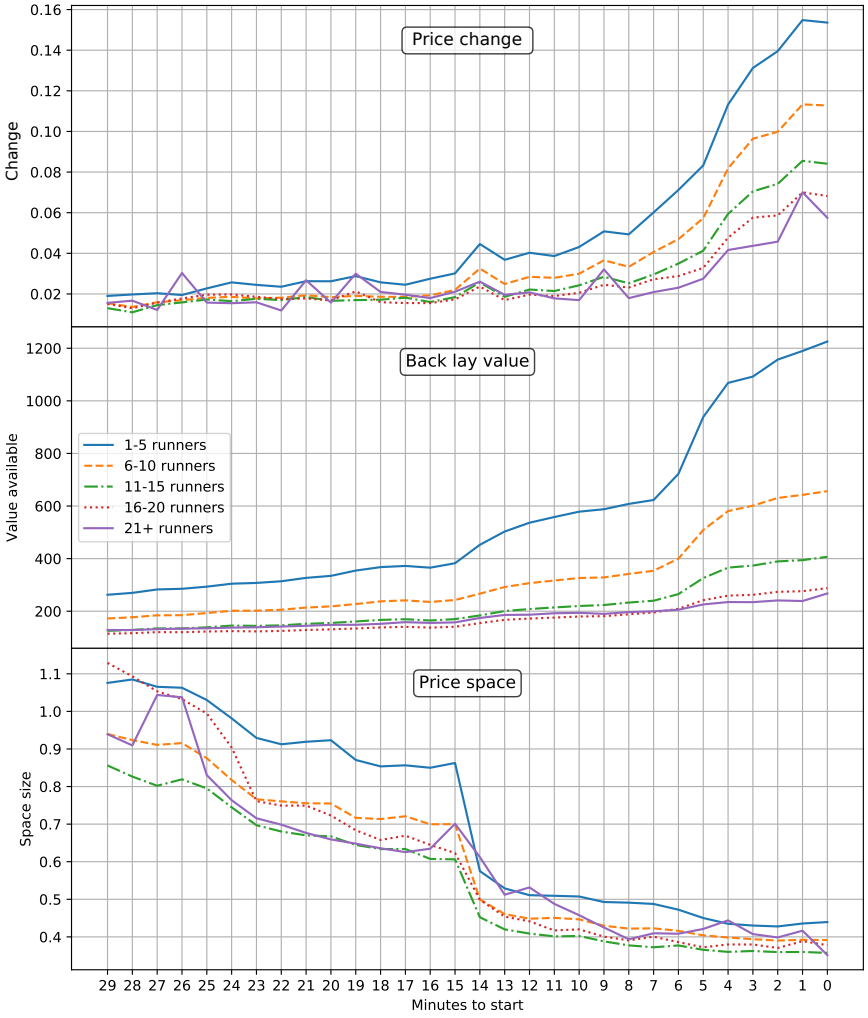


Figure 4.2: Evolution of basic market indicators in time, categorized and averaged by the count of runners.

Most of the examined features show the obvious: not all markets behave the same. This fact should be considered at the time of development of the strategies. The categorization of market dynamics could help with the specialization of individual predictors or picking some profitable subset of markets. Another finding is that markets change their properties during the pre-race time. Interval before 15 minutes to start should not be taken into account due to poor

market conditions. The later phase (15 minutes and less to start) should be split into several sub-intervals as all of the explored indicators are changing during this time.

4.2 Importance of RR features

Features that are presented in the dataset can be divided into two categories, namely market features and race-runner features (RR features). Market features describe the exchange process - price in time, the value of bets, total matched, etc. RR features describe the real event background - race type, class, distance, winning, runner age, weight, previous performance, jockey, trainer, etc. This section describes how valuable are the RR features for a price movement prediction.

4.2.1 Prediction task

The prediction task is a simple classification. The goal is to predict movement between a pair of time points, opening time and closing time. Three classes are considered:

- `BackLay` - for price moving down (increasing win probability),
- `LayBack` - for price moving up (decreasing win probability),
- `NoBet` - for no movement, or small movement which would lead to a loss.

The interval between opening and closing time is called prediction distance and can be changed for various conditions of the analysis.

4.2.2 Analysis

For this analysis, we fixed the closing time to 10 seconds before the race starts. Using the race start time (0 seconds) is not applicable, because as the race starts, the market gets frozen for a few seconds. For the opening time, we used 22 different uniformly distant options from range $[900 - 60]$ seconds before the start, hence prediction distance operates in the range $[890 - 50]$. We use multiclass logistic regression as the prediction model.

For prediction, we used three different datasets. Each of them extends the previous one.

1. **No RR data** - input features are only market data (see 3.3.2).
2. **Basic RR data** - extends the previous No RR data by race features and runner features (see 3.3.1).

3. **Ranked RR data** - extends previous Basic RR data by ranking runner features. Horse features are not so informative without knowledge of the other participants in the race. Ranking tries to exploit mutuality between individual runners in a race. Two simple ranking methods were applied on some runner features. **RankTop** - order of runner based on the observed feature. **RankMeanDiff** - difference from mean of observed feature.

Runner ID	OFR	OFR RankTop	OFR RankMeanDiff
102155981	83.0	2.0	8.4
102155982	65.0	5.0	−9.6
102155983	65.0	5.0	−9.6
102155984	84.0	1.0	9.4
102155985	76.0	3.0	1.4

Table 4.1: Example of OFR feature ranking application in the race with 5 runners.

Total of 66 (3×22) models were evaluated. The observed prediction score and profit per trade are presented in the following figure.

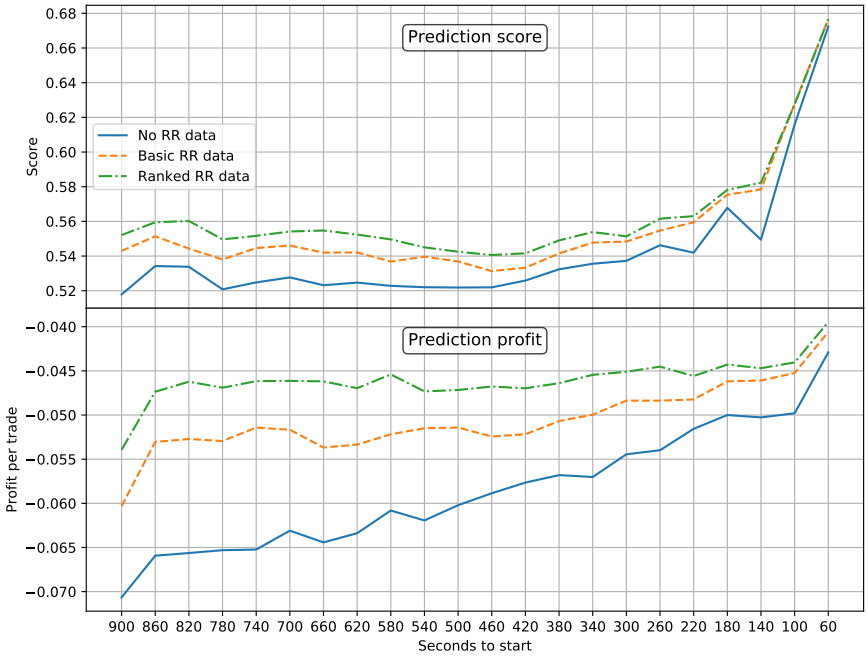


Figure 4.3: How race and runner features influences prediction.

The figure reveals that RR data, especially the ranked ones have a significant impact on model performance. Models with ranked features were always the best, both in score and profit metric. As the prediction distance decreases, all models get higher accuracy and the difference between them is vanishing out. That happens because the number of `NoBet` cases is increasing in the end. Unfortunately, the profit increase is not so significant.
Further feature importance analysis shows that ranked features were always more important than the non-ranked original values.

We conclude this section with a finding that a prediction on a longer time period should always be supported with RR data, additionally ranking techniques that include the relation between individual runners such as RankTop, RankMeanDiff can be used to improve the predictions even further.

4.3 Trade timing

For any betting strategy, it is critical to know when to open and close the trade. The combination of those two parameters along with the collection interval (the timespan which is used to collect the data served as the input for prediction) can suggest the best behavior for applying the trade.

The following analysis uses several values for those two parameters. The data collection interval (input window) was fixed for 200 seconds with 10 seconds granularity (20 inputs). Data of higher granularity was not needed for this part of the work. The prediction model is the same as in the previous section.

4.3.1 Analysis

Calculated heatmaps in figure 4.4 show that rather than the general gold combination of parameters, there is a well observable relation. Decreasing the opening time and shortening prediction interval improve the results in both metrics. That happens because as the market gets more active, even a short interval can lead to predictable price movement. In the early phase of the market nothing fundamental happens, the price movements hence appear only on longer prediction distances. Due to this fact white area is not included in the analysis, because more than 99.75 % (order of units in absolute numbers) of model outputs were `NoBet` cases (strongly defensive models) and hence the average profit was not relevant.

Further investigation pointed out how different window parameters influence input importance. Longer prediction interval models prefer the latest inputs from the collection interval. Shorter still relies mostly on the latest inputs, but on average puts more importance on the older data. In other words, models with shorter prediction distance try to deduce the future from the

whole observed input window. Models with longer prediction distance almost ignore what happened in the whole input window, they mainly exploit the latest inputs.

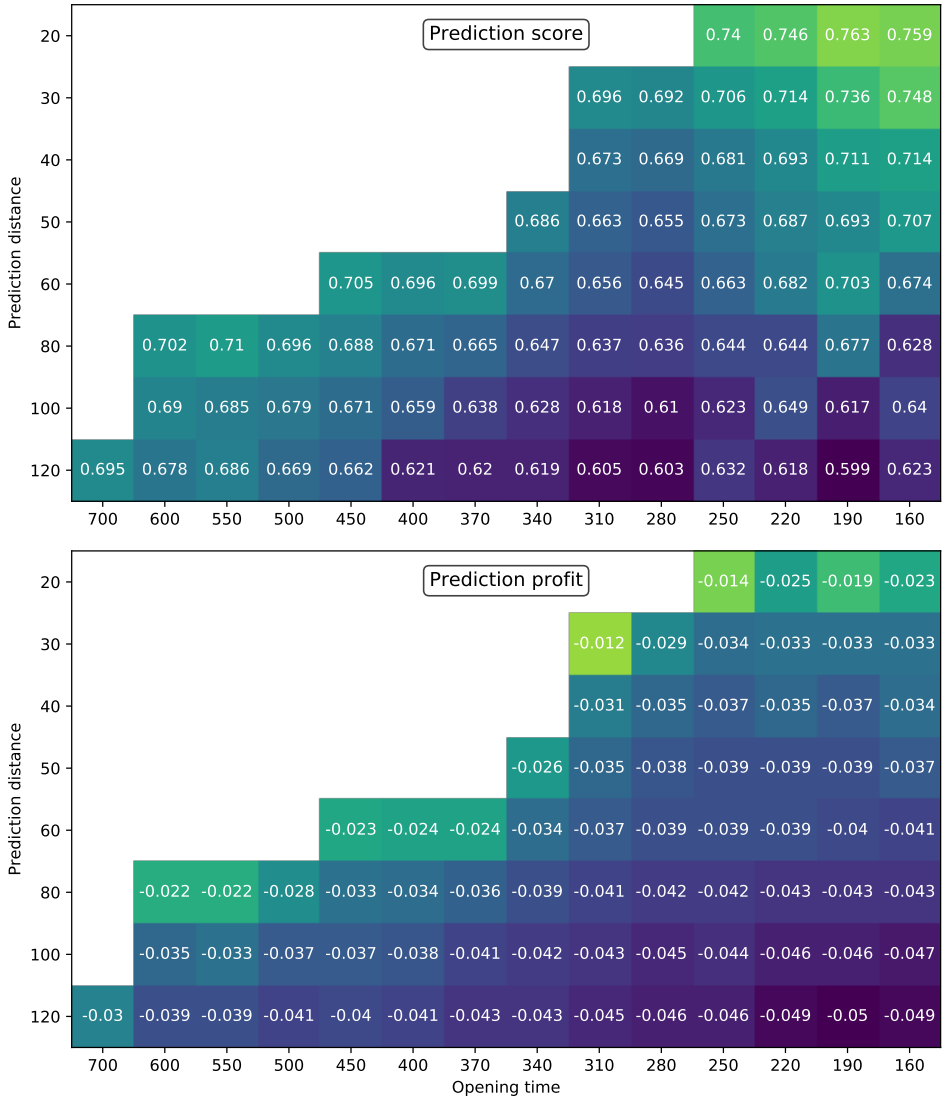


Figure 4.4: Heatmaps exploring basic trade timing parameters.

We conclude this section with a finding that for each opening time there is a different prediction distance achieving the best results. Moreover, the observed results point out the following relationship - lower opening time prefers lower prediction distance.

Chapter 5

Basic strategies

In this chapter, we describe what is a betting strategy and how it is constructed. Comparison metrics for different strategies will also be introduced. Finally, the two main betting concepts will be constructed and evaluated.

5.1 What is strategy

The strategy can be understood as a complex structure with 3 basic parts.

- **Criteria** - definition of the strategy itself and the way how it should be applied to the market. What kind of data will be collected, when and how can the trade be opened and closed. How many times can the trade be applied per individual runner in the race - single-trade, multi-trade. Criteria clarify all the basic properties.
- **Prediction** - According to the previously defined criteria, an appropriate dataset is generated. The dataset consists of many trading scenarios. Key components of this part are the choice of the prediction task, target value, and a loss function. A suitable machine learning (ML) algorithm is then trying to approximate the underlying function - mapping input features to the desired output value.
- **Execution** - the execution part applies the prediction model in practice. The main task is dealing with long-term wealth allocation - i.e. bet size, possible exit options, confidence rules restrictions, etc.

Moreover, in some cases, individual parts can interfere with others. For example, the prediction part could include the execution part via predicting bet amount and optimizing directly the long term profit. Changing individual parts of the strategy can influence overall evaluation. A more capable ML algorithm or appropriate target value can improve profitability. The same holds for wealth allocation, unit bet could be outperformed by ML optimized bet size.

5.2 Prediction target for ML models

The main interest of a prediction is obviously the price. As the market offers back and lay sides, the situation gets a little more complicated. At the start of a trade, it is necessary to determine which type of price-change is expected. Simply said price-change can either be an upward or a downward movement.

- **Downwards** - (decreasing decimal odds, increasing win probability) in this case, `BackLay` trade has to be applied - enter market with a back bet, then exit with a lay bet.
- **Upwards** - (increasing decimal odds, decreasing win probability) in this case, `LayBack` trade has to be applied - enter market with a lay bet, then exit with a back bet.

To exit a trade without a loss, it is needed to reach the close side with at least the same price as on the opening side. `BackLay` profit can be calculated via formula 5.1, `LayBack` profit via formula 5.2, both of them consider unit opening bet amount.

$$profit = \left(\frac{\text{open back price}}{\text{close lay price}}\right) - 1 \tag{5.1}$$

$$profit = 1 - \left(\frac{\text{open lay price}}{\text{close back price}}\right) \tag{5.2}$$

The following figure 5.1 shows an example of price evolution in the last 10 minutes of the pre-race time. Both back and lay prices are displayed.

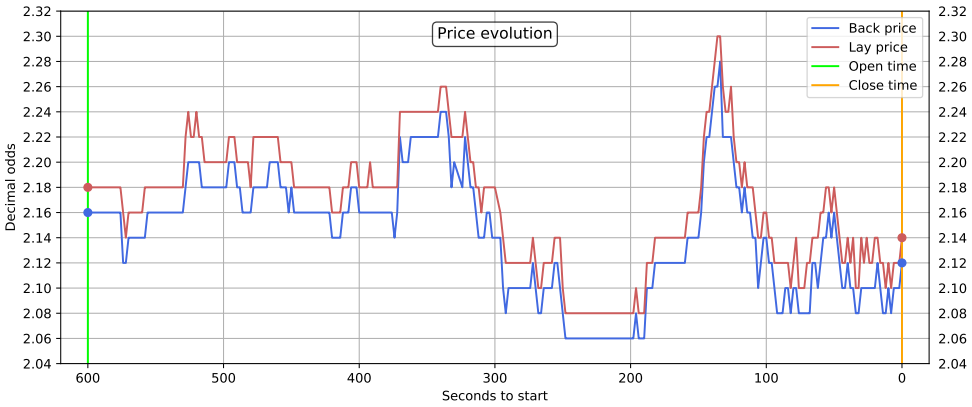


Figure 5.1: Example of back and lay price evolution in pre-race time.

5.2.1 Price after time interval

Strategy can have explicitly specified opening and closing time, the object of a prediction is then price after the specified interval. The ratio between opening and closing price is used as a prediction target to ensure the identical range of target values. Formulas for both price directions are together with an example values from previous figure described in 5.3 and 5.4.

$$BackLay = \left(\frac{\text{open back price}}{\text{close lay price}}\right) - 1 = \left(\frac{2.16}{2.14}\right) - 1 = \mathbf{0.0093} \tag{5.3}$$

$$LayBack = 1 - \left(\frac{\text{open lay price}}{\text{close back price}}\right) = 1 - \left(\frac{2.18}{2.12}\right) = \mathbf{-0.0283} \tag{5.4}$$

Regression models estimate directly this ratio, classification models estimate only negativity or non-negativity of the ratio. The provided example shows that tiny profit can be taken in `BackLay` case, while the `LayBack` case ends up in a small loss. The advantage of this target is that in the execution part there are only three possible decisions - `BackLay`, `LayBack` and `NoBet`. There is no benefit from predicting movement size because the overall profit will not be affected. The disadvantage is that right at closing time the close price may not be ideal.

See figure 5.1, few seconds before the close time higher profit was achievable due to lower lay price. Even higher profit was possible at approximately 220 seconds before the start. This consideration naturally leads to the following question: is there a way to exploit prices during the whole trading interval?

5.2.2 Price inside time interval

Strategy criteria can also specify explicit opening time together with variable closing time. In this case, the prediction target is the price inside the time interval. The main advantage is that the predictor is learning the best price achievable during the whole time interval, not only the price after that interval. On the other hand, if the predicted price is overestimated, the potential profit can easily turn into a substantial loss.

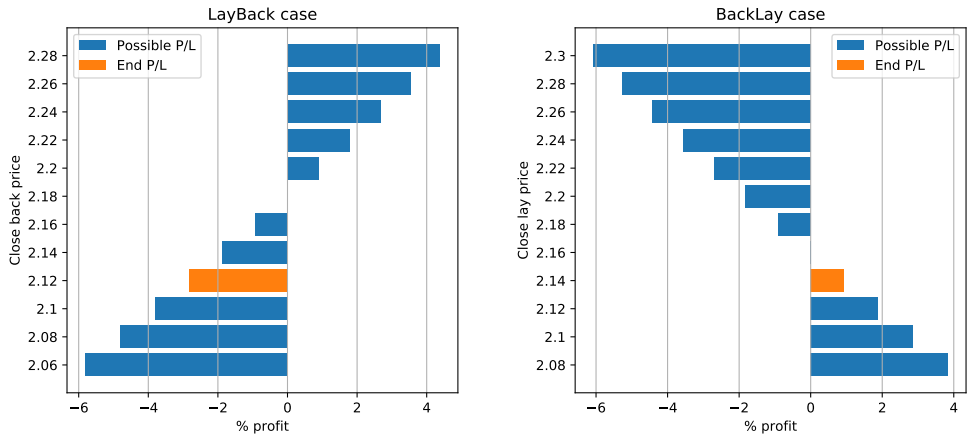


Figure 5.2: Possible profit and loss inside time interval.

Figure 5.2 shows all the possible closing back prices for `LayBack` case and all possible closing lay prices for `BackLay` cases (prices taken from figure 5.1). Orange color shows the price at the end of the observed period. The potential profit relative to the opening bet amount is placed on the x axis. In the previous target, where is predicted the end price, is profitable only BackLay case and just very slightly. Whereas both LayBack and BackLay variants are profitable in this case. Moreover the potential profit during this period is larger than the end profit. Formulas 5.5 and 5.4 for both cases are very similar to the previous formulas. Only the closing price was replaced by the best observed price.

$$BackLay = \left(\frac{\text{open back price}}{\text{best lay price}}\right) - 1 = \left(\frac{2.16}{2.08}\right) - 1 = \mathbf{0.0385} \tag{5.5}$$

$$LayBack = 1 - \left(\frac{\text{open lay price}}{\text{best back price}}\right) = 1 - \left(\frac{2.18}{2.28}\right) = \mathbf{0.0439} \tag{5.6}$$

From the properties of this prediction target, we can see that only the regression task can be applied here. It is very important to find a balance between the highest profit and the lowest risk. If the best possible price is taken as a target value, a low prediction error can lead to a loss. One of the possible solutions could be an asymmetric loss function designed in order to minimize overestimation of price. Another possible solution is to modify the predicted value to make our decision more robust to the error of overestimation.

5.3 Comparing strategies

For a detailed comparison of designed strategies, several metrics are proposed. Some of them are taken from [12].

- **Possible trades** - the count of total possible trades.
- **Profitable ratio** - the ratio of total possible profit trades to the total possible trades. A profitable trade is a trade where it is possible to achieve a profit.

$$\text{Profitable ratio} = \frac{\text{Profitable trades}}{\text{Possible trades}}$$

- **Perfect profit** - perfect profit is the total profit produced in case the strategy wins the highest possible profit in each profitable trade.
- **Placed trades** - the count of total placed trades.
- **Useful trades** - ratio describing how many of the placed trades were in the profitable trade set.
- **Useless trades** - ratio describing how many of the placed trades were outside of the profitable trade set.
- **AV win trade** - the average size of the winning trade (positive profit).
- **AV loss trade** - the average size of the loss trade (negative profit).

$$\text{Profit} = \text{Trade payout} - \text{Open bet value}$$

- **ROI** - return on investment evaluates the efficiency of an investment.

$$ROI(\%) = \frac{\text{Profit}}{\text{Total investment}}$$

- **SE** - strategy efficiency measures how efficiently a trading strategy converts the potential perfect profit into realized trading profit.

$$SE(\%) = \frac{\text{Profit}}{\text{Perfect profit}}$$

Useful trades and Useless trades metrics are used specifically to compare the inside time strategies and their ability to detect valuable trading opportunities.

5.4 Naive approach

We start with a very simple strategy that uses static conditions. The idea is based on the assumption of the trend remaining. If between two consecutive market states there is a price movement, which satisfies a predefined condition, it is then assumed that in the immediate future market state, this "trend" will remain.

5.4.1 Strategy description

The naive approach strategy goes state by state in the given price development of a single runner. Those states are enumerated as t (time), and the time between them is a strategy hyper-parameter Δ_t with the lowest possible value of 2 (original data granularity). The second hyper-parameter of this strategy is the set of conditions to satisfy before a trade is opened. There are two operations realized in each observed state.

1. **Evaluate opened trade** - if a trade (BackLay or LayBack) was opened in the previous state, it gets closed and evaluated.
2. **Open new trade** - check for the satisfaction of predefined conditions given to the prices in the previous and current state. If the conditions are satisfied, the appropriate trade is opened.

For this naive approach we propose two sets of conditions which will be tested separately:

1. **Change** - basic, benevolent set of conditions. If an increasing trend in back price is detected (given the previous state back price) - open LayBack trade. If a decreasing trend in lay price is detected (given the previous state lay price) - open BackLay trade.
2. **Profitable change** - extended, a more strict set of conditions. If the BackLay or LayBack trade theoretically opened in the previous state could be closed with profit in the current state, open such trade in the current state.

The first set of conditions checks just for some change. Second set checks for such change which makes a profit.

5.4.2 Example application

Let us assume that we try to apply the second set of conditions to price development provided in the following figure 5.3. The trade gets opened only in step t_3, because only there a possible trade was detected (LayBack case). In step t_4 this trade gets closed and evaluated, but as can be seen, the direction of development has changed and trade ended in a loss.

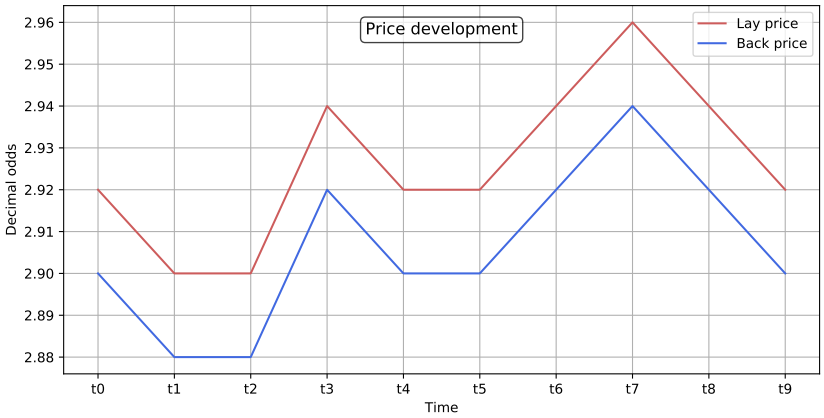


Figure 5.3: Example of price development.

5.4.3 Results

Naive approach results						
Set of conditions	Change			Profitable change		
Δ_t	16	32	64	16	32	64
Possible trades	7312	3646	1840	7312	3646	1840
Profitable trades	0.15	0.26	0.35	0.15	0.26	0.35
Perfect profit	42	45	41	42	45	41
Placed trades	6518	3030	1322	4189	2076	900
Useful trades	0.09	0.16	0.23	0.11	0.20	0.26
Useless trades	0.91	0.84	0.77	0.89	0.80	0.74
AV win trade	0.037	0.050	0.063	0.038	0.051	0.064
AV loss trade	−0.052	−0.062	−0.081	−0.051	−0.064	−0.089
ROI %	−3.702	−3.595	−3.799	−3.372	−3.333	−3.902
SE %	−571.6	−242.4	−121.8	−334.6	−154.0	−85.14

Table 5.1: Naive approach results.

Table 5.1 summarizes experiments provided in this naive approach section. All experiments were far from profit, but some of them performed better than others. The “Change” set of conditions was more offensive (the number of placed trades was higher), but the results (ROI and SE) were generally worse compared to “Profitable change“ set of conditions. Anyway, in both cases the ratio of useful and useless trades was very unbalanced, hence none of “naive” strategies was effective.

5.5 Strategy division

Building a more effective strategy may need some deeper reflection. As the heatmaps in section 4.3 show, for each prediction distance there exists a different opening bet time providing the highest prediction score. Opening bet time closer to a race start works best with short trades, opening bet time more distant from race start works best with longer trades. We will further refer to the two options as - **short trade** and **long trade**.

Previously we introduced two main concepts of the prediction target. Both of them will be used in the proposed strategies: **after time** and **inside time**.

The prediction task depends on the prediction target choice. In the after time case it can be both **classification** (CLF) and **regression** (REG). In the inside time target, it can be only **regression**. For classification, we have two cases (place a bet, do not place a bet). Regressed value represents the following:

- ratio between opening and closing price in the after time target case,
- ratio between opening price and best possible closing price in the inside time target case.

All of the mentioned variants and approaches are summarized in figure 5.4.

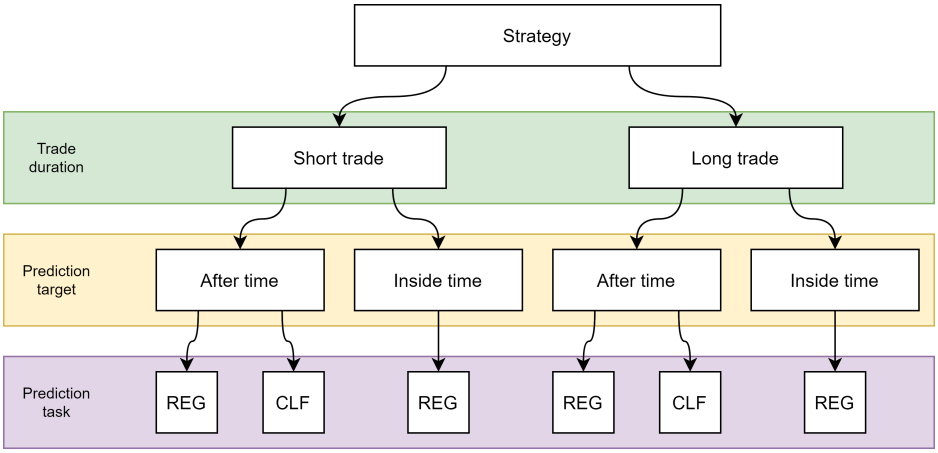


Figure 5.4: Composition of basic proposed strategies.

5.5.1 Prediction models

For both classification and regression tasks, we chose the following models based on the `scikit-learn` python library.

- **Classification** - RandomForestClassifier (RFC), GaussianNB (GNB).
- **Regression** - GradientBoostingRegressor (GBR), Ridge.

5.6 Long trade strategies

In the long trade case, only one trade per runner is applied. That is because the market is not active long enough for more long trades. Based on the explored importance of RR features in the previous chapter, it is expected that those features could have a positive influence on the overall result as the prediction distance, in this case, is long.

5.6.1 Input data

For strategy evaluation, we used 10 000 randomly selected races from the whole dataset. Converted to the number of runners, the number of observations is about 150 000. We use both market and RR features as input. The input features carry information about the current price development and additional features describing the race and their participants.

- **Market data** - basic indicators were chosen from the market data - best available back and lay price together with the total matched money. This triplet is obtained each minute during the observation interval. Input interval starts 30 minutes before the race starts, the end depends on the opening time, i.e. the moment when the trade position is opened.
- **RR data** - the most valuable features come from the RR data as shown in our analysis in section 4.2. The included features are:
  - **Race features** - number of runners, class, winning, distance, TV.
  - **Runner features** - OFR, TSR, RPR, weight, tips, signs, jockey ratio, trainer ratio, owner ratio.

  Furthermore, the runner features are processed over RankTop and RankMeanDiff ranking methods instead of being used directly, both methods were already described previously in section 4.2.2.

5.6.2 Additional parameters

Some additional parameters were introduced in the sense of increasing the chance of capturing some profitable market conditions. Those parameters are trade opening and trade closing time, for this strategy instances they are $\{1200, 900, 600\}$ and $\{120, 60, 2\}$ seconds to race start.

5.6.3 Evaluation process

The original dataset was split into the train and validation sub datasets. Each of the proposed prediction models was used with all suitable prediction targets (6 possibilities). Each of these possibilities was evaluated with all possible

combinations of additional parameters (9 possibilities). See figure 5.4 for details. A total of 54 scenarios were evaluated. Moreover, all hyper-parameters of individual models were selected using a grid search, i.e. trying all combinations of model-specific parameter values from a given discrete set. Accuracy was used for the classification models and mean average error (MAE) for the regression models as the score function.

After the training process, separate test data were used to describe the resulting performance. The model prediction is fed into the evaluator, which calculates final values for the pre-defined metrics.

5.7 Short trade strategies

In the short trade case, more trades per single runner are applied. As the market provides enough activity for a period, where multiple short trades can be executed. Because the prediction distance is at most one minute, no RR data were used in those models, because the analysis in section 4.2 proposed that their positive influence on the prediction score for short time trade is negligible.

5.7.1 Sliding window

Sliding windows is a technique, which is generating multiple learning observations from the price development of a single horse in a single race. The purpose is to capture short-term price drifts and try to estimate an immediate market reaction.

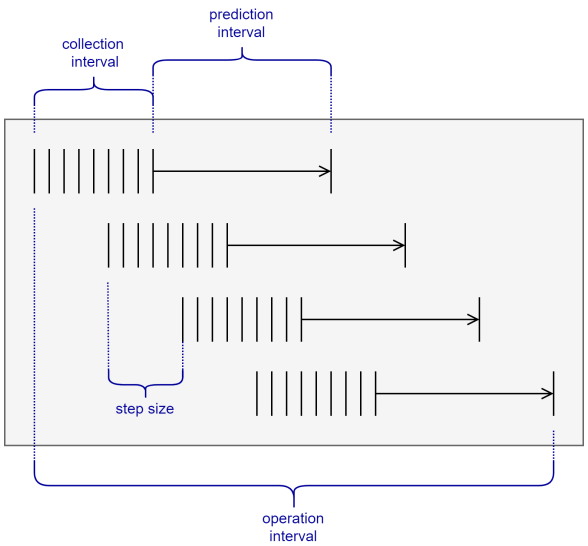


Figure 5.5: Schema of the sliding window technique.

- **Collection interval** - time interval in which the data is collected as the input for the estimator. Trade opening bet can be placed at the end of the collection interval.

- **Prediction interval** - time interval in which the closing price is predicted (both after time and inside time targets). Opened trade must be closed inside this interval.

- **Step size** - specifies the size of the shift between two consecutive windows, i.e. the density of windows inside operation interval.

- **Operation interval** - time interval in which the windows are generated.

5.7.2 Input data

For strategy evaluation, we used 200 000 sliding windows selected from more than 6 000 random runners. As the input features, we used only the market data.

- **Market data** - from the market data, we used similar features as in the long trade case. They were extended by a feature describing the ratio between the total back and lay value available. The data density is much higher, the original 2-second granularity is used, i.e. the densest what we have.

5.7.3 Additional parameters

Some additional parameters were again introduced to increase the chance of spotting profitable conditions. First is the prediction interval of the sliding window, tested values were $\{30, 60, 90\}$. The second parameter was the start of the operation interval, two values $\{600, 300\}$ were tested.

5.7.4 Evaluation process

The evaluation process is very similar to the previous section, only the number of scenarios has decreased. All combinations of possible prediction algorithms and prediction targets (6 possibilities) were tested with all possible combinations of additional parameters (6 possibilities). A total of 36 scenarios were evaluated. See figure 5.4 for details.

5.7.5 Other conditions

Both prediction targets, their models, scoring functions, etc. are the same as in the previous section.

5.8 Results and findings

Best long trade models						
Prediction target	After time				Inside time	
Prediction task	Regression		Classification		Regression	
Prediction model	GBR	Ridge	RFC	GNB	GBR	Ridge
Opening time	1200	600	900	1200	600	600
Closing time	120	2	2	60	2	2
Possible trades	18738	18372	18534	18738	18372	18372
Profitable trades	0.37	0.40	0.40	0.38	0.75	0.75
Perfect profit	1084	1265	1341	1206	2317	2317
Placed trades	481	748	3673	6286	18352	18366
Useful trades	0.52	0.56	0.42	0.42	0.75	0.75
Useless trades	0.48	0.44	0.58	0.58	0.25	0.25
AV win trade	0.137	0.124	0.167	0.142	0.104	0.103
AV loss trade	−0.144	−0.146	−0.200	−0.202	−0.208	−0.208
ROI %	0.212	0.519	−4.60	−5.75	−4.21	−4.26
SE %	0.094	0.307	−12.56	−29.98	−33.3	−33.8

Table 5.2: Best long trade models.

Best short trade models						
Prediction target	After time				Inside time	
Prediction task	Regression		Classification		Regression	
Prediction model	GBR	Ridge	RFC	GNB	GBR	Ridge
Prediction interval	30	60	30	90	30	30
Operation int. start	600	300	600	600	600	600
Possible trades	19362	19468	19362	19330	19362	19362
Profitable trades	0.23	0.26	0.13	0.23	0.20	0.20
Perfect profit	315	324	120	315	176	176
Placed trades	122	16	210	3045	4202	2160
Useful trades	0.42	0.25	0.25	0.31	0.37	0.34
Useless trades	0.58	0.75	0.75	0.69	0.63	0.66
AV win trade	0.042	0.049	0.058	0.088	0.013	0.010
AV loss trade	−0.041	−0.071	−0.089	−0.071	−0.124	−0.124
ROI %	−0.614	−4.112	−5.225	−2.171	−0.063	−0.068
SE %	−0.237	−0.202	−9.143	−20.98	−1.519	−0.840

Table 5.3: Best short trade models.

Two tables 5.2 and 5.3 summarizes experiments provided in this chapter. From each combination of prediction target, task, and model was selected as the model with the best achieved SE. The SE was also used as the main metric to compare the proposed strategies.

The **classification** models failed in all provided conditions, they have also the worst ROI results over all models. The **regression** models performed better, some of them were even slightly profitable.

The **inside time** target variant did not hit expectations. As the table 5.2 shows, for long trade, the number of placed bets almost equals the total number of possible bets. On the other hand, for a short trade, the results were significantly better. The difference between average win and loss is very high compared to the other targets. None of the inside time models were able to learn how expensive the overestimation of the best price is in the sense of overall profit.

The **after time** target combined with the regression task performed relatively well. Models were betting carefully, especially in the short trade case. In the long trade case, a small profit was achieved. It utilizes about 0.3 % and 0.09 % of potential profit, so there is still quite a large space for improvement.

In the provided experiments were also tracked the influence of additional parameters. 600 seconds before the race start was the preferred opening time for the long trade models. As the best closing time was picked value 2, but the difference of results between tested closing times is very low.

Additional parameters of **short trade** strategies pointed out, that smaller prediction interval offers lower perfect profit, but generally can be better exploited. Also, the strategies relatively strictly preferred windows with wider operation interval.

Chapter 6

Neural networks strategies

One of the planned branches of this thesis was building a strategy that would use a neural network (NN) predictor. Part of the works studied in the research chapter also tried to use neural networks. The previous experiments pointed out some promising combinations of parameters. Neural network models are generally more capable than previously used models, this capability could better capture the hidden relationships in the data. [3]

We propose 3 strategies with specific neural network architectures.

1. Basic NN with typical architecture - few hidden fully connected layers using standard loss function, trying to predict the price after the time interval.

2. The second strategy uses very similar conditions to first architecture, but the prediction target is price inside the interval. As this target was not very successful in the previous chapter, we provide a custom loss function, which could increase the resulting score.

3. The last proposed strategy tries to exploit the ability of CNN to identify shapes in the time series data.

All NN models are using `Keras` python library with `TensorFlow` backend.

6.1 Basic NN - after time, long trade

This strategy tries to predict the price after time interval in the long trade duration using the regression task. The mentioned conditions should be a good starting point, as the same were the most promising in the previous chapter. For the opening and closing time (additional parameters), we used values 600 and 2.

6.1.1 Input data

Input data are similar to the experiment in the previous chapter. Similarly to the previous experiments: 10 000 randomly selected races are used, which corresponds to 150 000 runners.

6.2 Custom loss NN - inside time, long trade

The second strategy tries to predict the best price inside the time interval in the long trade duration using the regression task. Because the expectations from this prediction target were not met in the previous chapter, we try to design a specific loss function instead of using standard MAE.

6.2.1 Input data

Input data are the same as in the Basic NN experiment.

6.2.2 Loss function

Typical loss function describes the relationship between predicted and desired value, no other values are involved. But for more complex situations Keras offers a utility for implementation of a specific loss function, which can access data outside the network itself. Such models have a set of input and output vectors and an objective function. In the following example, both inputs *best_price* and *end_price* are together with *pred* used to calculate overall loss.

```
def custom_loss():
    mask = K.less(pred, best_price)

    loss_basic = K.square(best_price - pred)
    loss_additional = K.square(end_price - pred)
    loss_total = loss_basic + loss_additional * mask

    return K.mean(loss_total)
```

The proposed loss function consists of two partial losses. The basic loss penalizes the difference between the predicted best price and the actual best price. If the predicted best price overestimates the actual best price, the second additional loss is activated. The second loss penalizes the difference between the predicted best price and the actual end price. In practice this means, that model should rather underestimate the best price in an effort to avoid a big loss.

6.3 CNN - after time, short trade

The last strategy tries to predict the price after the time interval in the short trade duration using the regression task. Dense market data used as input in short trade can be seen as multinomial time series data. This could be utilized by the convolution version of neural networks, which are suitable, inter alia, for time series. [13]

6.3.1 Input data

Input data are the same as in section 5.7, i.e. 200 000 sliding windows from more than 6 000 races. Each window consists of 16 sequential time moments (rows) and in each moment, there are 14 features (columns). The sample of a single window is presented in the following figure 6.1.

B1P	B1V	L1P	L1V	B2P	B2V	L2P	L2V	B3P	B3V	L3P	L3V	Matched	Ratio
0.000000	0.0	0.000000	0.0	0.000000	0.0	0.000000	0.0	0.000000	0.0	0.000000	0.0	0.0	0.452055
-0.007937	5.0	-0.005040	-53.0	-0.008201	-19.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	586.0	0.582633
-0.005376	66.0	-0.005040	-67.0	-0.002825	-69.0	-0.004884	-41.0	-0.002922	60.0	-0.004735	-37.0	604.0	0.680851
-0.005376	66.0	-0.005040	-67.0	-0.002825	-69.0	-0.004884	-41.0	-0.002922	60.0	-0.004735	-37.0	744.0	0.680851
-0.005376	66.0	-0.005040	-67.0	-0.002825	-69.0	-0.004884	-41.0	-0.002922	60.0	-0.004735	-37.0	744.0	0.680851
-0.007937	-4.0	-0.005040	-67.0	-0.008201	-19.0	-0.004884	-41.0	-0.008479	-41.0	-0.004735	-37.0	744.0	0.552980
-0.007937	-4.0	-0.005040	-67.0	-0.008201	-19.0	-0.004884	-41.0	-0.008479	-41.0	-0.004735	-37.0	744.0	0.552980
-0.005376	16.0	-0.005040	-67.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	1138.0	0.560261
-0.005376	16.0	-0.005040	-73.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	1571.0	0.571429
-0.005376	16.0	-0.005040	-74.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	2271.0	0.573333
-0.005376	16.0	-0.005040	-74.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-6.0	-0.004735	-37.0	2613.0	0.577558
-0.005376	16.0	-0.005040	-74.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	2695.0	0.573333
-0.005376	16.0	-0.005040	-74.0	-0.005557	-66.0	-0.004884	-41.0	-0.005747	-9.0	-0.004735	-37.0	2719.0	0.573333
-0.010417	-4.0	-0.009775	-59.0	-0.010761	-103.0	-0.011671	49.0	-0.011123	7.0	-0.011322	-120.0	2719.0	0.466192
-0.012821	-46.0	-0.009775	-59.0	-0.013242	-89.0	-0.011671	49.0	-0.013684	9.0	-0.011322	-120.0	2719.0	0.411765
-0.012821	-46.0	-0.009775	-59.0	-0.013242	-89.0	-0.011671	49.0	-0.013684	9.0	-0.011322	-120.0	2837.0	0.411765

Figure 6.1: Example of the sliding window before normalization.

All columns, except the last one, are defined in section 3.3.2. The last column represents the ratio between the money available on back and lay side. The ratio is calculated via the following formula.

$$\text{Ratio} = \frac{\sum_{i=1}^{3} \text{B}i\text{V}}{\sum_{i=1}^{3} \text{B}i\text{V} + \sum_{i=1}^{3} \text{L}i\text{V}} \tag{6.1}$$

6.3.2 Architecture data

Figure 5.5 shows a simplified architecture of the network. Convolution layer with four filters, with two different sizes (yellow and red color), is applied on the input matrix. Application of a filter creates a feature vector - a result of the convolutional layer. Global max-pooling operation is applied in the next layer - from each feature vector, the maximum value is taken as the feature corresponding to this particular filter. The idea is to take the most important feature from each feature map. The pooling is then followed by a fully connected layer which has to process pooled features into the final network numeric output. [14]

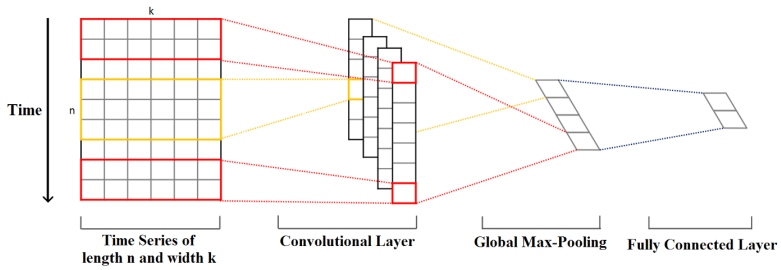


Figure 6.2: Simplified architecture of CNN network. [15]

6.4 Tuning hyper-parameters

Hyper-parameters are parameters, which cannot be inferred in the learning process, but their values define the learning process itself. Each of the proposed networks has many hyper-parameters that affects the result of network learning. To obtain the best results, multiple values for those hyper-parameters should be tested. [3, 13]

6.4.1 Important hyper-parameters

- **Network hierarchy** - the number of hidden layers and the number of neurons in each of them. Deeper and denser networks usually can better generalize the underlying problem as their capacity is higher. On the other hand, they also tend to overfit.

- **Activation function** - each neuron output is processed over the activation function. Activation often squeezes the raw neuron output into some range, which helps to decide whether the neuron is active or inactive.

- **Optimizer** - is an algorithm that determines the direction and size of an update in network parameters in order to reduce the overall loss. Some algorithms also offer additional parameters, such as initial learning rate.
- **Dropout** - is a regularization parameter, which helps to avoid overfitting. The dropout rate specifies how many connections between random neurons in a certain layer are dropped during the learning.
- **Number of epochs** - specifies the length of the learning process. Can be replaced by early stopping criteria.
- **Batch size** - specifies the number of training observations used in one learning iteration. Smaller training sets can be processed quickly, but the direction of the network parameter updates can easily catch the local noise. Larger sets provide precise direction, but the processing time is much longer.
- **Weight initializer** - defines how to set the initial weights of neurons in the layer. Some kind of initialization can prevent the exploding or vanishing gradient and also help with faster convergence into optimum.
- **Convolution settings** - includes both hierarchy of convolution layers and convolutional settings in each of those layers. Settings are the number of filters, kernel size, strides and padding. Convolution layers are often followed by pooling layers, which help reduce the spatial size of the representation.

6.4.2 Tuning

The number of hyper-parameters is much higher than in the models used in the previous chapter. Using only grid-search is therefore not practically applicable, possible searching would take too much time, as the complexity of search space has exponential growth with the number of hyper-parameters. To find the right trade-off between reasonable searching time and achieved result, knowledge from resources was combined with decent grid-search. In practice, this means that complex parameters, such as the architecture of network or convolution layers, were inspired by other works in the time series prediction field. The same holds for the hidden layers activation function, where the ReLU was chosen. The remaining parameters such as optimizer, weight initialization, batch size, or dropout rate were selected using grid-search. [16]

Tested values for hyper-parameters in grid-search:

- **optimizer** $\in$ {Adagrad, Adadelta}
- **learning rate** $\in \{0.01, 0.001\}$

- **batch size** $\in \{16, 32\}$
- **dropout** $\in \{0.2, 0.35, 0.5\}$
- **weight initialisation** $\in$ {glorot normal, he normal}

6.5 Optimized networks and hyper-parameters

The following table 6.5 shows the final configuration of hyper-parameters for all proposed networks - Basic NN, Custom loss NN and CNN.

Final configuration of neural networks			
Model	Basic NN	Custom loss NN	CNN
Optimizer	Adagrad	Adadelta	Adadelta
Activation	ReLU	ReLU	ReLU
Dropout rate	0.35	0.35	0.2
Batch size	32	16	32
Weight init.	he normal	he normal	he normal
Learning rate	0.001	0.001	0.001
Stride			1
Padding			valid
Architecture [1]	Dense(128) Dense(64) Dense(32) Dense(16) Dense(8) Dense(4) Dense(2) Dense(1)	Dense(128) Dense(64) Dense(32) Dense(16) Dense(8) Dense(4) Dense(2) Dense(1)	Conv1D(100,6) Conv1D(100,4) Dropout() MaxPooling() Conv1D(160,3) Dropout() MaxPooling() Flatten() Dense(128) Dense(64) Dense(32) Dense(16) Dense(8) Dense(4) Dense(2) Dense(1)

Table 6.1: Final configuration of neural networks.

The overall architecture of the CNN network is based on architecture provided in [17]. The convolution part is followed by several fully connected layers, which are usually used in CNN architectures, for example in AlexNet. [14]

Basic NN and Custom loss NN both use narrow architectures with few hidden

[1] A dropout layer is placed between each pair of consecutive dense layers. We omit this in the table for clarity.

layers. Dimensional data is not presented in their inputs.

By the grid-search is then provided final optimization of remaining hyper-parameters in given networks. Note that in all networks, the *he normal* initializer and the 0.001 learning rate provided the best results.

6.6 Results and findings

Best NN models			
Prediction task	Regression		
Prediction target	After time	Inside time	After time
Model	Basic NN	Custom loss NN	CNN
Possible trades	18372	18372	19362
Profitable trades	0.40	0.75	0.23
Perfect profit	1265	2317	315
Placed trades	2897	17694	1257
Useful trades	0.38	0.78	0.43
Useless trades	0.62	0.22	0.57
AV win trade	0.117	0.035	0.031
AV loss trade	−0.138	−0.195	−0.037
ROI %	−4.114	−1.562	−0.572
SE %	−9.412	−11.91	−2.282

Table 6.2: Summarization of experiments provided in this chapter.

Conditions used in the Basic NN model are the same as in the previous chapter, where 0.307 % SE was achieved using the Ridge model. Anyway, the current model ended in a loss, the number of placed trades was significantly higher, as the model is more offensive. The average winning trade is lower and the ratio between useful and useless trades has changed for the worse.

The second model using custom loss tries to exploit the application of different prediction target. Compared to models in the previous chapter, the number of placed trades is lower, the ratio of useful trades is slightly improved and the best SE has moved from −33.3 % to −11.91 %. The additional penalty for overestimation meets its purpose. Despite this, there is still space for improvement as the overall numbers are negative.

The last proposed model, using convolution layers, should outperform models from the previous chapter. The best SE achieved in short trade with after time target was −0.237 %. SE of the proposed CNN model is −2.282 %, but other indicators, such as the ratio of useful trades or average win trade show better results than basic models. The value of negative SE is caused by a

high number of placed trades, if we compare the ROI value, the CNN model is slightly better.

Unfortunately, none of the presented models provided a profitable strategy. The only noticeable improvement was in the Custom loss NN model. However, the numbers are still far from the edge of profit.

Conclusion

This thesis focused on the design of pre-match betting strategies using machine learning algorithms. The golden (nice to have) output should be a strategy, which can generate long-term profit in real world conditions. The main output is the gradual digging into the problem - analysis of the relation between race and market features, searching for optimal strategy criteria, trying various strategy properties, etc. Finally a set of strategies should be proposed according to provided analysis.

We started with the betting world, main principles and participants were introduced. Then decent research of work in similar topics was done. The research pointed out that one of the favorite sports is horse racing due to its specific behavior in the pre-race phase. Also, several specific properties and needs of horse racing markets were described, i.e. quick development of market conditions, the problem of the loss function selection, or categorization of the market by race properties.

The following work was data pre-processing, formatting, and reshaping. Remotely stored data were downloaded and processed into an appropriate file format, which was suitable for fast read/write operations. All kinds of features were converted into the correct data format. Wrongly stored or missing data were removed or corrected if it was possible, some big issues led to event removal. Then all the related data were reshaped into several dataframes using the tabular shape for further analysis - race, horse, market data. Although this part looks simple, the invested time was relatively big, finding all the anomalies and specifics is tedious.

When the data was ready, the analysis could start. Three main aspects were explored - dynamic of market conditions, the importance of non-market data and optimal trade timing. The first aspect pointed out is that approximately 15 minutes before the prescribed race start time, market conditions significantly improves for betting. The second aspect reveals the importance of

non-market data, as they achieved a better score than market data itself. The effect of importance was even stronger if features of individual horses contained some information about their relative rank to other horses in the race. The last aspect studied the optimal trade timing - trade opening and closing time. The analysis identified that the optimal trade duration depends on the opening time. As the market activity gradually increases with approaching start time, shorter trades are preferable.

In chapter 5, strategy definition and comparison metrics were introduced and strategies were proposed. As each strategy is formed by several parts (trade duration, prediction target, prediction task), for each part we proposed certain possibilities. Moreover, each of the proposed strategies is parametrized by additional parameters. All of these options were designed with respect to the conclusions from the previous analysis. The resulting framework tested each possible combination of mentioned parts and parameters. In total, 90 strategies were tested and compared. Summarizing table in section 5.8 showed two profitable strategies, both using very similar conditions, the better one achieved 0.307 % SE, other configurations had a negative profit.

The last chapter tried to increase performance on a subset of proposed strategies. To achieve better performance, more capable ML models were used - artificial neural networks. Three strategies from the previous section were selected and for each of them, we proposed a different ANN architecture. Two models used basic deep neural network architecture, third used CNN architecture trying to detect features in the time series data. Unfortunately, only one of three models increased the previously achieved score and all of them had a negative profit.

Future work

The number of possible future approaches are numerous. The first group of them could be a continuation of existing work. For example, the extension of the set of provided strategies, improvement of ML models, finding new loss functions, etc.
The second group of approaches is trying completely different methods. For example, it could be reinforcement learning, where the agent would receive the state of the world (bank balance, market data, race data) and respond with some action (place a bet, close/hold opened bet, do nothing), then he receives a positive or negative reward (market profit/loss, a penalty for inactivity).

Bibliography

[1] Mike Huggins. *Flat racing and British society, 1790-1914 : a social and economic history*. London Portland, OR: Frank Cass, 2000. ISBN: 0714680451.

[2] The Telegraph. *The gambler who bet on himself*. May 2005. URL: https://www.telegraph.co.uk/finance/2916471/The-gambler-who-bet-on-himself.html.

[3] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and echniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, Inc, 2019. ISBN: 978-1492032649.

[4] Dominic Cortis. "Expected Values and Variances in Bookmaker Payouts: a Theoretical Approach Towards Setting Limits on Odds". In: *The Journal of Prediction Markets* (2015).

[5] Mark Davies et al. "Betfair.com: Five technology forces revolutionize worldwide wagering". In: *European Management Journal* (2005). ISSN: 02632373. DOI: 10.1016/j.emj.2005.09.008.

[6] The Guardian. *Free bets mean you can clean up as bookies meet their match*. July 2010. URL: https://www.theguardian.com/money/2010/jul/24/free-bets-bookies.

[7] Egon Franck, Erwin Verbeek, and Stephan Nuesch. "Inter-market Arbitrage in Sports Betting". 2009.

[8] Rui Gonçalves et al. "Deep learning in exchange markets". In: *Information Economics and Policy* (2019). ISSN: 01676245. DOI: 10.1016/j.infoecopol.2019.05.002.

[9] Ivars Dzalbs and Tatiana Kalganova. "Forecasting Price Movements in Betting Exchanges Using Cartesian Genetic Programming and ANN". In: *Big Data Research* (2018). ISSN: 22145796. DOI: 10.1016/j.bdr.2018.10.001.

[10] Øyvind Norstein Øvregård. "Trading "in-play" betting Exchange Markets with Artificial Neural Networks". MA thesis. NTNU, 2008.

[11] Ilia Zaitsev. *The Best Format to Save Pandas Data*. Mar. 2019. URL: https://towardsdatascience.com/the-best-format-to-save-pandas-data-414dca023e0d.

[12] Polyvios Tsirimpas. "Specification and Performance Optimisation of Real-time Trading Strategies for Betting Exchange Platforms". PhD thesis. Imperial College London, Apr. 2014, pp. 62–65.

[13] Antonio Gulli. *Deep learning with TensorFlow 2 and Keras : regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API*. Birmingham: Packt Publishing Ltd, 2019. ISBN: 1838823417.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* (2017). ISSN: 15577317. DOI: 10.1145/3065386.

[15] Yoon Kim. *Convolutional Neural Networks for Sentence Classification*. 2014. arXiv: 1408.5882.

[16] Hsieh CH. "Detection of Atrial Fibrillation Using 1D Convolutional Neural Network". In: *Sensors (Basel)* (2020).

[17] Nils Ackermann. *1D Convolutional Neural Networks in Keras for Time Sequences*. Sept. 2018. URL: https://medium.com/p/3a7ff801a2cf.

Chapter 7

Acronyms

UK - United Kingdom

IRL - Ireland

ML - Machine Learning

NN - Neural Network

DNNC - Deep Neural Network Classifier

LSTM - Long Short-Term Memory

CNN - Convolution Neural Network

ANN - Artificial Neural Network

JSON - JavaScript Object Notation

CSV - Comma Separated Values

RR - Runner Race

CLF - Classification

REG - Regression

MAE - Mean Average Error

MSE - Mean Squared Error

ReLU - Rectified Linear Unit

Chapter 8

Contents of enclosed CD

```
src.zip ..... implementation sources
text ..... the thesis text directory
    thesis.pdf ..... the thesis text in PDF format
```