Imperial College
London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Serve Strength Ranking Approach to Modelling of Professional Tennis Matches

Author:
Erik Grabljevec

Supervisor:
Dr William Knottenbelt

Submitted in partial fulfillment of the requirements for the MSc degree in Computing Science of Imperial College London

September 2015

Abstract

Mathematical tennis modelling has received a lot of attention in the past few years. Main motivation for the developments in this area is the widespread of betting on tennis matches, with millions worth of bets being placed on the top matches. Another important aspect is prediction of the game duration, used in media. Furthermore tennis modelling is interesting as it overlaps with other areas of the research such as finance or machine learning.

Currently built models for tennis match prediction are divided into two groups; models that use player rankings and hierarchical markov chain models. Models that rank players lack flexibility and insight. They don't provide much more than the estimate of the probability of players winning a match. On the other hand we have markov models that provide a lot of insight and possibilities for enhancements once the input values are estimated.

In my thesis I will build a model that uses a ranking system to estimate the values used in a hierarchical markov chain model. We will evaluate feasibility of different ranking systems used in such an approach, evaluate different models and attempt to design a specific ranking system for our model. Finally, we provide ways to enhance such a model and demonstrate possible applications.

Acknowledgements

I wish to express my gratitude to my thesis advisor Dr. William Knottenbelt, for his inspiring guidance through this work.

I would like to thank Paolo Puggioni and Stratagem for all the help, support and code provided.

Finally, I would like to thank my girlfriend, Sara Pohl, for her unconditional love and support through the studies and writing of this thesis. Without her support this thesis could not have been completed.

Contents

List of Figures

Chapter 1

Introduction

Tennis is a very interesting sport to model from the mathematical perspective. Only two players are engaged in one match, game scoring has a hierarchical structure and there is an extensive match history data available.

Modelling of a tennis match is interesting from several perspectives. A good model can provide an insight into dynamics of a tennis match. Media has a great interest to estimate duration of the match in order to organize their schedule. To continue, modelling can be used for coaching purposes like tracking of player's career or to spot player's weaknesses. Finally, there is a rising betting market, where both bet makers and bettors have a great interest in estimating winning probabilities of the players.

There are two major approaches to tennis modelling. First approach is to use estimated winning probabilities of both players winning a point on serve. These probabilities are then used in a hierarchical markov chain model to calculate probabilities of players winning a match. Markov chain model and its implementation in Excel was explored by Barnett [7], [8]. On the other hand, different ranking systems were employed to estimate players' skills. One such strong attempt was done by McHale and Morton [18].

Hierarchical markov chain models are superior in tennis modelling, since they provide a really good insight in a match. They recursively build from one point played up to the entire match. This enables modification when using such an approach, like done by Madurska [17] who does a set by set analysis. Nonetheless, one can analyse other aspects of the game, besides winning probability. Duration of the match can be estimated or players' serve vs return strengths can be observed. Ranking models on the other hand don't provide such versatility.

Creating a new model is very straightforward if one is given a good framework to implement it in. In start of the project Stratagem provided such a framework. Inspired by that framework, we build a very lightweight framework that supports fast

implementation and analysis of new models. It builds local database and provides all main functions one might need to implement and analyse new models. This new framework will be released publicly at the end of the project, speeding up the creation of new models. For guide on how to use it, one should refer to the appendix C.

In this work we try a new approach to tennis modelling, utilizing both ranking systems and the markov chain model. Ranking system is used to compare serve strengths to return strengths of tennis players. Using this ranking system we estimate probabilities of players winning a point on serve. These values are then fed into the markov chain model.

We attempt development of a specific ranking system that would be tailored to our model. This ranking system is used in our base model and optimized. Once this ranking system is optimized, ranking system acts very similar to Elo ranking system and provides only a small margin over Elo ranking.

Using data generated with our base model, we build a graphical tool that presents form of the players through time. With assist of a graphical tool we analyse our base model to find room for the optimization. We present an improved model and give guidelines for application and future work.

1.1 Aims and objectives

The aims and objectives of this study are:

- Provide a new ranking system that would be most suitable for tennis modelling. Implement this ranking system in a simple to use Python library
- Develop Python framework that is easily installed and used to develop new tennis ranking models
- Take an innovative approach to tennis modelling. Provide a new direction for development of tennis models
- Develop graphical tool that will use model data to get insight in player's form and career development
- Give guidelines for the future research based on this thesis

1.2 Project outline

Chapter 2: Presents background on hierarchical markov chain model and ranking systems.

Chapter 3: This chapter presents a new ranking system, modified Glicko ranking system. It combines Elo ranking system with Glicko ranking system.

Chapter 4: Working framework and the evaluation metric are presented. This is followed with the presentation of our base model using modified Glicko ranking system. Finally, we show graphical tool that is used for analysis of model and players.

Chapter 5: In this chapter we spot pattern and problems in our base model. We suggest possible improvements that are later used in the improved model.

Chapter 6: We apply observations to improve our base model.

Chapter 7: Here we give guidelines for application of our model supported with example applications.

Chapter 8: In final chapter we summarize what was achieved and what could be done in the future.

Chapter 2

Background and Related Work

2.1 Markov model

Tennis is a very interesting sport to model from the mathematical perspective. Tennis match's scoring system is split in sets, each set in games and each game in several points. Such a structure is great for a hierarchical mathematical modelling. Reader can find simple presentation of tennis scoring system on wikipedia website [4].

Barnett [7], [8] did extensive research on the hierarchical tennis match modelling. In this section I will present how such a model can be used to calculate the probability of a player winning a match. Once we have the probabilities of both players winning a point on serve, we can recursively calculate probabilities of players winning the game, tiebreaker game, set and match. This makes hierarchical markov model very important for our model and it will be presented in depth in the next subsections.

It is important to note that we are using an $i.i.d.$ (independant identical distribution) for all points played. Madurska [17] has proven that this preposition doesn't hold completely and his work can be applied to improve any hierarchical model.

2.1.1 Modelling a game

We start the analysis by modeling one game of a set. A game is won by the player who first wins four or more points with a two point advantage over the opponent. Markov model of a game can be seen on figure 2.1.

Lets p denote probability of player A winning a point against player B and $1 - p$ probability of B winning against A, when player A is on serve. Probability of player A winning the game when serving against player B when score is (a, b) is denoted

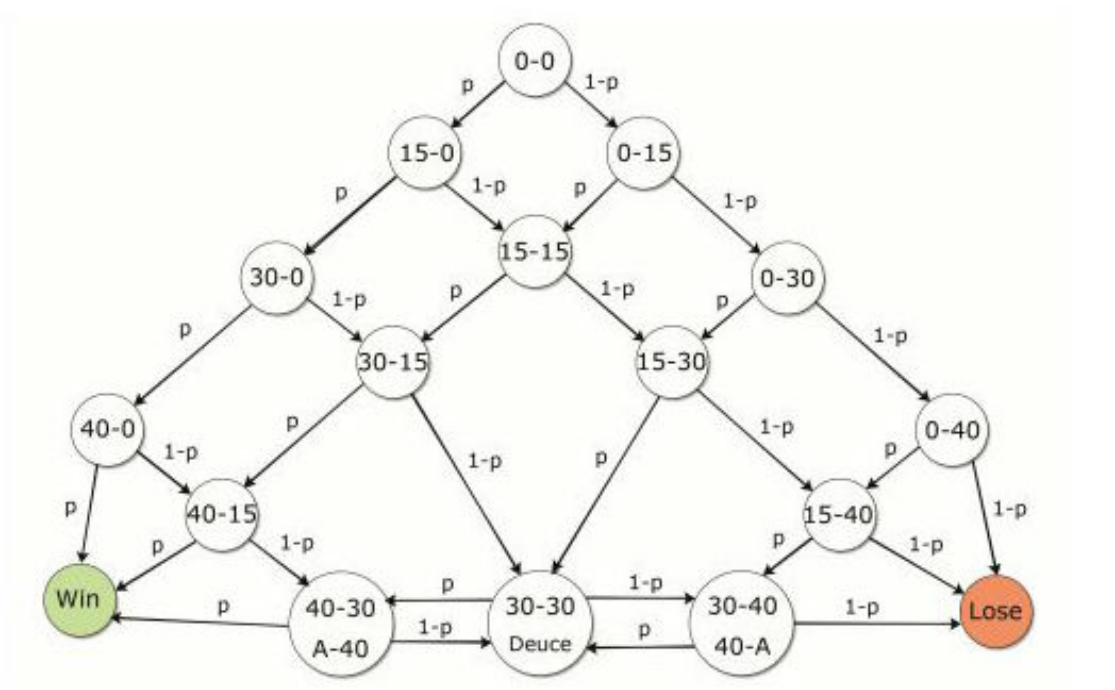

Figure 2.1: Graph of a Markov model for a tennis game (source [6]).

as $P(a, b)$. $P(a, b)$ can be written recursively as:

$$P(a, b) = pP(a + 1, b) + (1 - p)P(a, b + 1) \tag{2.1}$$

The boundary values are $P(a, b) = 1$ if $a = 4, b \leq 2$, $P(a, b) = 0$ if $b = 4, a \leq 2$. A problem arises when the score is $(3, 3)$. In that case our problem presented with the above boundaries recurses indefinitely. Nonetheless, we can calculate $P(3, 3)$ explicitly.

There are 4 possible scenarios for the next 2 games. Two of them bring us into the same tied situation and the other two finish the game. We can write $P(3, 3)$ with the next equation:

$$P(3, 3) = p^2P(5, 3) + 2p(1 - p)P(4, 4) + (1 - p)^2P(3, 5) \tag{2.2}$$

Using $P(3, 3) = P(4, 4)$, $P(5, 3) = 1$ and $P(3, 5) = 0$ we get:

$$\begin{aligned} P(3, 3) &= p^2 + 2p(1 - p)P(3, 3) \\ (1 - 2p(1 - p))P(3, 3) &= p^2 \\ P(3, 3) &= \frac{p^2}{p^2 + (1 - p)^2} \end{aligned} \tag{2.3}$$

		B				
		0	15	30	40	game
	0	0.62	0.46	0.27	0.10	0.00
	15	0.76	0.61	0.41	0.18	0.00
A	30	0.88	0.77	0.60	0.33	0.00
	40	0.96	0.92	0.82	0.60	
	game	1.00	1.00	1.00		

Table 2.1: Probabilities of player A winning the game from various scorelines against player B.

Table 2.1 shows probabilities for all possible pairs of scores (a, b), where $p = 0.55$.

Analogically to p and P; q is the probability of player B winning a point against player A when B is on serve and Q is the probability of player B winning the game.

2.1.2 Modelling a tiebreaker game

Before we can model a set, there is another special game that needs to be modeled. When set score is $6 - 6$, set is decided by playing a tiebreaker game. When playing tiebreaker one player serves first point, the second serves the next two. Afterwards, players exchange serve every two points played. First player to reach 7 points with a 2 point advantage wins the tiebreaker game, as well as the set.

Modelling a tiebreaker game is very similar to the one of the normal game, found in equation 2.1. To differentiate between normal game and tiebreaker game we add superscript T to probabilties when referring to a tiebreaker game. Using the same notation as before we write:

$$\begin{aligned} P^T(a,b) &= pP^T(a+1,b) + (1-p)P^T(a,b+1), \text{if } (a+b) \div 2 \pmod 2 = 0 \\ P^T(a,b) &= qP^T(a,b+1) + (1-q)P^T(a+1,b), \text{if } (a+b) \div 2 \pmod 2 = 1 \end{aligned} \tag{2.4}$$

The boundary values are $P(a,b) = 1$ if $a = 7, b \leq 5$ and $P(a,b) = 0$ if $b = 7, a \leq 5$. Using the same logic as in equation 2.2 we calculate:

$$P(6,6) = \frac{p(1-q)}{p(1-q) + q(1-p)} \tag{2.5}$$

2.1.3 Modelling a set

Let P^S denote the probability of player A winning a set. When player A has the first serve we model the set with the next two equations:

$$\begin{aligned} P^S(a,b) &= P * P^S(a+1,b) + (1-P) * P^S(a,b+1), \text{if } (a+b) \pmod 2 = 0 \\ P^S(a,b) &= Q * P^S(a,b+1) + (1-Q) * P^S(a+1,b), \text{if } (a+b) \pmod 2 = 1 \end{aligned} \tag{2.6}$$

The boundary values are $P^S(a,b) = 1$ for $a = 6$ and $b \leq 4$ and $P^S(a,b) = 0$ for $b = 6$ and $a \leq 4$. We are left with unknown boundary condition $P(5,5)$. When the set score is $5-5$ there are several possible scenarios, however only two of them bring player A to victory. Either player A wins two games in a row or player A wins one and loses one of the next two games and then wins the tiebreaker game. We can write this with an equation as:

$$P^S(5,5) = PQ + (PQ + (1-P)(1-Q))P^T \tag{2.7}$$

2.1.4 Modelling a match

Having probabilities for player A winning a set against player B, modelling a match is straightforward. We denote the probability of player A winning a match as P^M and write the next equation:

$$P^M(a,b) = P^S P^M(a+1,b) + (1-P^S)P^M(a,b+1) \tag{2.8}$$

The boundary values for the best of three match are $P^M(a,b) = 1$ if $a = 2$, $b \leq 1$ and $P^M(a,b) = 0$ if $b = 2$, $a \leq 1$. The boundary values for the best of five match are $P^M(a,b) = 1$ if $a = 3$, $b \leq 2$ and $P^M(a,b) = 0$ if $b = 3$, $a \leq 2$.

2.1.5 Tennis match formulae

Equations presented in the previous sections were solved by O'malley [19]. Once we have obtained the probabilities of players winning a point on serve (p and q) we can use equations in this subsection to calculate probabilities of players winning a match.

Probability of winning a game is:

$$P = \frac{p^4 * (15 - 4 * p - 10 * p^2)}{p^2 + (1-p)^2} \tag{2.9}$$

Probability of winning a tiebreaker game:

$$P^T = P^T(0,0) = \sum_{i=1}^{28} A(i,1)p^{A(i,2)}(1-p)^{A(i,3)}q^{A(i,4)}(1-q)^{A(i,5)}d(p,q)^{A(i,6)}$$
$$d(p,q) = \frac{p(1-q)}{p(1-q)+q(1-p)} \tag{2.10}$$

Values for A can be found in appendix A.
Probability of winning a set:

$$P^S = \sum_{i=1}^{21} B(i,1)P^{B(i,2)}(1-P)^{B(i,3)}P^{B(i,4)}(1-P)^{B(i,5)} \times ((PQ + (P(1-Q)) + (1-P)Q)P^T)^{B(i,6)} \tag{2.11}$$

Values for B can be found in appendix A.
Probability of winning a match:

$$P^M = P^M(0,0) = (P^S)^2(1+2(1-P^S)), \quad \text{if best of 3}$$
$$P^M = P^M(0,0) = (P^S)^3(1+3(1-P^S)+6(1-P^S)^2), \quad \text{if best of 5} \tag{2.12}$$

2.1.6 Estimating values p and q

There are several approaches to estimating the probabilities of players winning a point on serve (p and q). Barnett [9] averages players' statistics over all opponents. This model has a very poor performance as it is biased in a sence that strong players tend to play against strong players more often and weak players against weak players. This implies that a strong and a weak player might have very similar stats but a large difference in their true skill level.

Currently the best approach was done by Knottenbelt [16], where common opponent approach is used to estimate input. Such a model provides strong estimates and is currently considered the state of art model for tennis modelling.

Towards the end of the article, Knottenbelt discusses the possibility of using multiple depths of common opponent. Considering the idea of multiple depths, we tried to imagine how infinite depth of common opponents would look like. How could we compare all the values indirectly? Our answer to this question was to estimate values p and q using a ranking system.

2.2 Ranking systems

In sports and in other competitive activities ranking systems are used to compare strengths of participants. Throughout the history different ranking systems were introduced. Often a system that awards points to competitors proportional to their achievments is used. For example golf players are awarded a number of points, depending on the level of a tournament and their placement. Those points will be kept by the end of the season.

In this work we are more interested in ranking systems that employ mathematical methods. Such ranking systems have their roots in chess and were first started by Harness [2] and later by Elo [3]. Chess ranking systems were thoroughly researched by Glickman [11], who proposed a new, improved ranking system ([13], [14]). Some of the latest work on ranking systems was done by Microsoft's research team, that created a new rating system Trueskill, which is mostly used for online gaming ([15]). All these systems not only compare players but also estimate the probability of one player winning over the other. Furthermore, for the needs of our later presented model, we would like to work with match results between 0 and 1 (instead of just win/loss).

In the following subsections I will explore different ranking systems and discuss possible modifications to suit our task at hand.

2.2.1 Elo ranking system

Elo ranking system was invented by Arpad Elo in order to rank chess players. It is well presented on wikipedia website [3].

Elo ranking is great for mathematical models as it not only ranks players but also returns probability of one player winning over another. This probability is estimated using logistic curve presented in figure 2.2. We can write the expected score of player A winning over player B as:

$$E_A = \frac{1}{1 + 10^{(R_A - R_B)/400}} \tag{2.13}$$

Where R_A is rating of player A and R_B rating of player B.

After a match between A and B is finished, with player A scoring S_A we would update ratings accordingly:

$$R_A^{'} = R_A + K(S_A - E_A) \tag{2.14}$$

Where K is a modification factor. For chess $K = 32$ is used, however this parameter should be optimized for a specific model used.

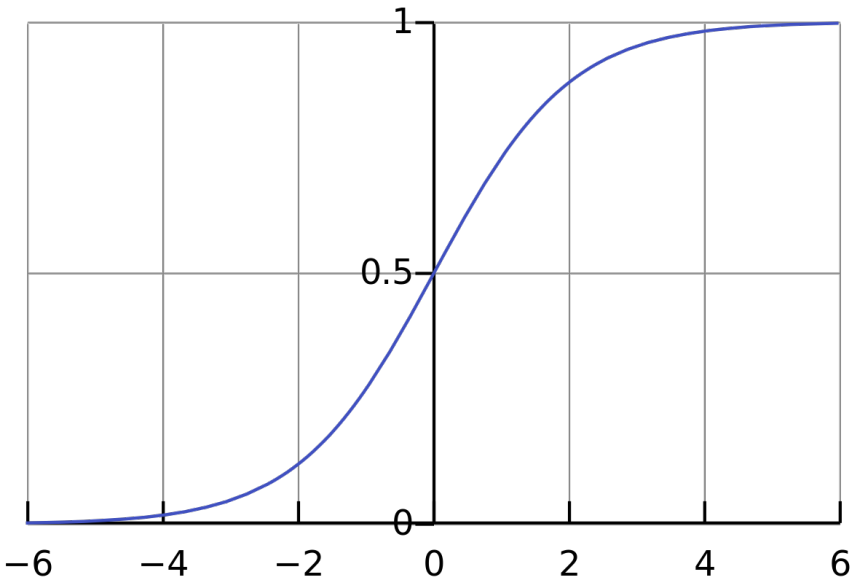

Figure 2.2: Logistic curve.

Now imagine that R_A would denote player A's serve ability and R_B player B's return ability. E_A would then denote exactly the probability of player A winning a point on serve, p. This will be the main idea of our new model presented in this work.

2.2.2 Glicko ranking system

While Elo ranking system has been proven very effective in practice, it has a major limitation. If player A has a rating of 1500 after 0 games or after 100 games it is treated the same by the system. If player A wins a game against player B with rating 1700, his rating will increase by the same value regardless of the number of games player A has played so far. In other words, Elo ranking doesn't take into account reliability of players' ratings.

This problem is tackled by Glicko [13], who adds another variable to player's rating, that is rating variability. This new approach was proven to be more effective. Glicko rating system splits games into rating periods. After each such rating period re-evaluation of all player ratings is done. Furthermore, Glicko rating can be used for results that are on interval between 0 and 1. So far Glicko ranking system seems ideal for the needs of this project. There are however two problems with this system.

Firstly, when a player hasn't played for a while his rating variability is reduced. This works very well for example in onine gaming, as players are often taking breaks from the game. However, tennis players aren't taking breaks from career as often. There is another problem with reducing player's rating variability when they are not

playing. It might appear that a player is inactive, while in fact he often plays in tournaments we are not collecting data for. Secondly, Glicko ranking system calculates rankings after each period, which is not as robust as we would like it to be for our analysis, especially for graphical representation.

In the beginning of the next section we are proposing a modification of Glicko ranking system to fit our needs. This new ranking system is truly more of an improvment of the Elo ranking system then modifiction of the Glicko ranking system.

2.2.3 Other ranking systems

There exist other ranking systems that weren't presented in this chapter. Most notable are Glicko2 and Trueskill ranking systems. Both are built from Glicko ranking system. Trueskill expands it to multiple players and uses Gaussian function instead of logistic. On the other hand Glicko2 ranking system tries to incorporate momentum of players' ranking.

Calculations used in these two ranking systems are rather complicated and since neither of them produced interesting results for our model we don't provide indepth explanations. If the reader is interested in how these two systems work, he can refer to [14] for Glicko2 and to [15] for Trueskill ranking system.

Chapter 3

Modified Glicko ranking system

As discussed in the previous chapter, Glicko [13], [14] upgraded Elo ranking system by adding another variable, reliability of player's rating. In Glicko ranking system each player is assigned two values, rating r and rating reliability RD. RD depends on the activity of a player. A player that regularly competes has a higher rating reliability compared to an inactive player. Glicko uses the following equation to calculate RD:

$$RD = \min\left(\sqrt{RD_{old}^2 + c^2 t}, 350\right) \tag{3.1}$$

Where t is the number of rating periods since the last competition and c is a constant that governs the increase in uncertainty over time.

We modify Glicko's approach by changing how RD is calculated. To calculate RD for a player that played m matches we use the following equation:

$$RD = \min\left(RD_{begin} - mRD_{step}, R_{end}\right) \tag{3.2}$$

We can tell that RD starts as a linear function and turns into a constant when it reaches RD_{end}. Figure 3.1 demonstrates RD as a function of the number of matches m for $RD_{begin} = 300$, $RD_{end} = 100$ and $RD_{step} = 20$.

Once we calculate RD we can modify rankings on match by match basis. To update the rating of player 1 after a match with player 2, we use equations 3.3.

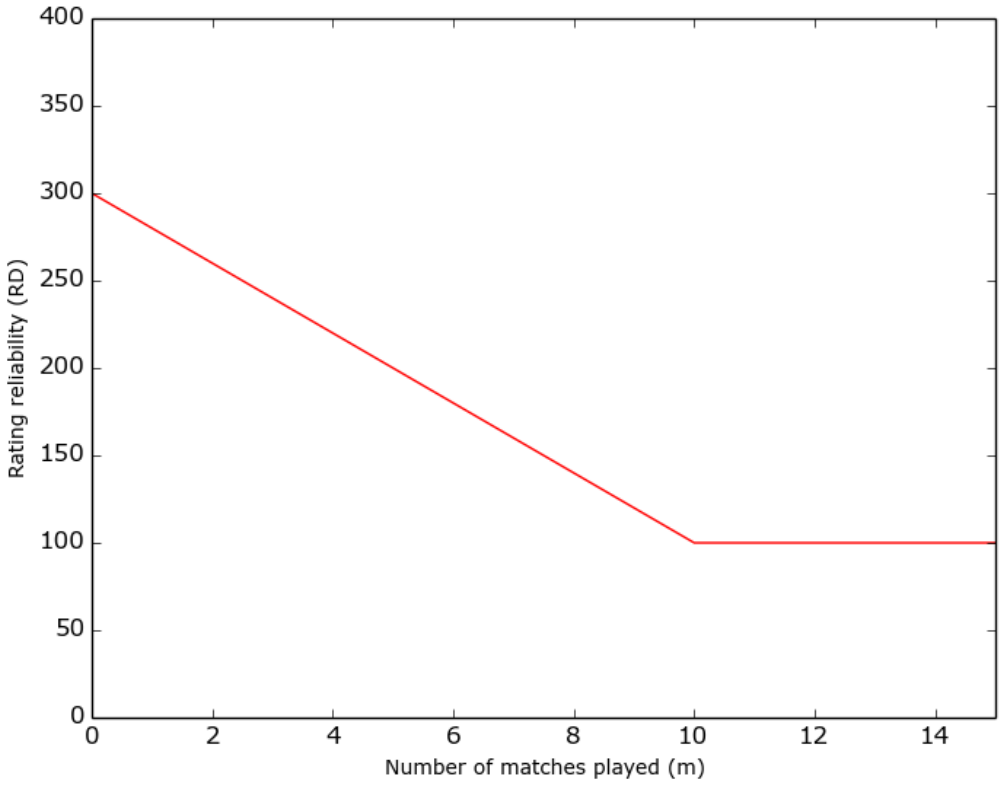

Figure 3.1: RD as a functon of the number of matches played m.

$$
\begin{aligned}
r_1' &= r_1 + \frac{q}{1/RD^2 + 1/d^2} g(RD_2)(s - E(s|r_1, r_2, RD_2)) \\
RD &= \min\left(RD_{begin} - mRD_{step}, RD_{end}\right) \\
q &= \frac{\ln 10}{400} = 0.0057565 \\
g(RD) &= \frac{1}{\sqrt{1 + 3q^2(RD^2)/\pi^2}} \\
E(s|r_1, r_2, RD_2) &= \frac{1}{1 + 10^{-g(RD_2)(r_1 - r_2)/400}} \\
d^2 &= (q^2 g(RD_2)^2 E(s|r_1, r_2, RD_2)(1 - E(s|r_1, r_2, RD_2)))^{-1}
\end{aligned}
\tag{3.3}
$$

There is another subtle advantage of Modified Glicko ranking system; we can modify parameters to fit the type of analysis we want to perform. If we want to perform either a career analysis or a tournament analysis, we can modify RD_{end} accordingly. Lower RD_{end} is better for career analysis, while higher can analyse changes on a smaller time scale.

3.1 Example calculation

In this subsection we present an example of use of Modified Glicko ranking system. We set parameters to $RD_{begin} = 300$, $RD_{end} = 100$ and $RD_{step} = 20$.

Player A has 20 games played with a rating of 1500. Player B has 6 games played with a rating of 1700. We can calculate both players' rating reliability:

$$\begin{aligned} RD_1 &= \min(300 - 20 * 20, 100) = 100 \\ RD_2 &= \min(300 - 6 * 20, 100) = 180 \end{aligned} \tag{3.4}$$

Players compete and player 1 scores 0.75 against player 2. We now update players' ratings. One can expect that player 1 will gain rating, while player 2 will lose rating. Moreover, the rating of player 2 will change more drastically since his rating is less reliable.

We calculate the rating change for player 1:

$$\begin{aligned} g(RD_2) &= \frac{1}{\sqrt{1 + 3q^2(RD_2^2)/\pi^2}} \\ &= \frac{1}{\sqrt{1 + 3 * 0.0057565^2 * 180^2/\pi^2}} \\ &= 0.868302275 \end{aligned}$$

$$\begin{aligned} E(s|r_1, r_2, RD_2) &= \frac{1}{1 + 10^{-g(RD_2)(r_1 - r_2)/400}} \\ &= \frac{1}{1 + 10^{-0.8683 * (-200)/400}} \\ &= 0.269006320 \end{aligned} \tag{3.5}$$

$$\begin{aligned} d^2 &= (q^2 g(RD_2)^2 E(s|r_1, r_2, RD_2)(1 - E(s|r_1, r_2, RD_2)))^{-1} \\ &= (0.0057565^2 * 0.895786^2 * 0.269 * (1 - 0.269)) * * - 1 \\ &= 203547 \end{aligned}$$

$$\begin{aligned} r_1' &= 1500 + \frac{0.0057565}{1/100^2 + 1/203547} * 0.8683 * (0.75 - 0.269) \\ &= 1523 \end{aligned}$$

To obtain the rating change of player 2 we repeat the above procedure and calculate $r_2' = 1629$. In this example player 1 won unexpectedly over player 2. Nonetheless player 1 didn't gain that much rating since his rating reliability is very reliable. On the other hand, the rating of player 2 changes drastically as his rating is unreliable, because of few matches played.

Chapter 4

Base model

We follow the suggestion of a new ranking system with a new model for tennis match prediction. In this chapter we present how we tackled this task, starting with how we set up the working framework and choose a criterion for model comparison. Later in the section the new model and a graphical tool visualizing its output are presented.

4.1 Working framework

As mentioned, our framework was inspired by Stratagem's. It has less functionality but it is simpler and easier to modify.

The database is built locally or on the selected server by running a script, which parses the data from excel spreadsheets. Data was obtained from two sources ([1], [5]). Data is loaded from the database using Pandas library.

We implement the model by deriving it from the class *TennisRankingModel* and implementing *run* method. Once we derive it from *TennisRankingModel* we run *analyse_ranking_model* to analyse the model and to produce a latex report, such as the ones found in the appendix B.

For a longer guide to our framework refer to the appendix C.

4.2 Model evaluation

To optimize the parameters in our models and to compare different models we need an evaluation metric. A Good metric should be easily calculated and usable in every model. We use the metric, used by Glickman [12] and McHale, Mortone [18]:

$$d_{ij} = -s_{ij} \log(p_{ij}) - (1 - s_{ij}) \log(1 - p_{ij}) \tag{4.1}$$

Where d_{ij} means error, s_{ij} score and p_{ij} estimated score for player j on match i. Error for model d is calculated as a sum over all d_{ij}.

We use the above model error to optimize model parameters. However, we also calculate the other two model strength predictors. The first one is the percentage of correct predictions calculated by the model. A model has calculated a correct prediction on a match if it predicted more than 50% probability for the winner of the match. The second evaluation criterion is how model performs against a betting exchange. Here we take the best odds from the four major exchanges. Since we get the data by interpolating two sources, some of the data on odds was unfortunately lost.

Model reports are stored in the appendix B. Errors are calculated for years from 2003 to 2015. For years 2012, 2013, 2014 and 2015 correct prediction percentage and betting against exchange ROI is calculated.

4.2.1 Evaluation example

Model name: Random
Error: 27996.43889
Bias: 0.49867
Sqme: 0.33102
Correct predict percentage:

2012	2013	2014	2015
0.49164	0.50727	0.49609	0.49212

Predict mean: 0.49678
Predict std: 0.00630
Predict 95% confidence interval: (0.47674, 0.51682)
Bets placed:

2012	2013	2014	2015
446	452	420	224

Betting ROI:

2012	2013	2014	2015
−0.12814	0.05381	−0.02936	−0.31058

ROI mean: -0.07440
ROI std: 0.11962
ROI 95% confidence interval: (-0.45509, 0.30628)

To get a basic feel for how the evaluation works, we evaluate a completely random model, which returns a random value from 0 to 1 to predict each match. On the previous page we present a report of such an evaluation. It starts with the main information about the model; the name, description and parameters used. This is followed by the training error, bias and square mean error. Next there is a section on prediction and at the end we test our model against a betting exchange.

Most results of a random model are expected. The model has bias and prediction close to 0.50. More interesting is betting ROI. While the random model is losing as expected with an average ROI close to exchange margin, there is a winning year with a 5% return. One should be cautious when evaluation strength of the model with a betting ROI. Nonetheless, we will present the betting results of the models, as this is a very universal metric, commonly used in the evaluation of tennis models.

In the appendix B one can find several reports. Some of them aren't discussed in this thesis but might be still interesting to the reader. We suggest the reader to look at the appendix B.2, where we present the report of the ad hoc implementation of the Barnett's model [9]. The decision to store all models' reports to an appendix was made so that we can methodically present several models.

4.3 Model

Using the ideas from the previous sections, we try to formulate a new model. This model uses a ranking system to estimate probabilities of players winning a point on serve. It feeds those probabilities in a hierarchical model (as done by Barnett [7]) to obtain probabilities of players winning a match.

In section 2.1 we have presented how one can calculate the probabilities of a player winning a match, given the probabilities of players winning a point on serve - denoted p and q.

In our model, we consider a tennis match as two seperate matches. Player 1 serving against player 2 and player 2 serving against player 1. With w_1 we denote the percentage of points player 1 won on serve and with w_2 for player 2. Each player is then assigned two ratings, serve strength and return strength. For player 1 we denote serve rating as r_1^s and return rating as r_1^r. Furthermore, each player is assigned a rating reliability RD.

We have the layout equations for modified Glicko ranking system in section 3. Using two of them, shown in the equation 4.2, we can now calculate the probabilities p and q.

$$
\begin{aligned}
E(s|r_1, r_2, RD_2) &= \frac{1}{1 + 10^{-g(RD_2)(r_1 - r_2)/400}} \\
g(RD) &= \frac{1}{\sqrt{1 + 3q^2(RD^2)/\pi^2}}
\end{aligned}
\tag{4.2}
$$

To calculate p, we will use ratings r_1^s and r_2^r and to calculate q, ratings r_2^s and r_1^r. We derive the following equations:

$$
\begin{aligned}
p &= E(s|r_1^s, r_2^r, RD_2) \\
q &= E(s|r_2^s, r_1^r, RD_1)
\end{aligned}
\tag{4.3}
$$

We can feed p and q values into markov chain model and calculate the probability of players winning a match. After the match is over, we can use the actual percentages of points won on serve to update the ratings of both players.

4.3.1 Example calculation

Lets take player 1 with serve rating $r_1^s = 1504$ and return rating $r_1^r = 1434$. For the second player we take $r_2^s = 1583$ and $r_2^r = 1491$. We set the same rating reliability to both players $RD_1 = RD_2 = 100$.

Should those two players play against each other, we can calculate the expected p and q using equation 4.3.

$$
\begin{aligned}
g(RD) &= \frac{1}{\sqrt{1 + 3q^2(RD^2)/\pi^2}} \\
&= \frac{1}{\sqrt{1 + 3 * 0.005^2(100^2)/\pi^2}} \\
&= 0.96404
\end{aligned}
$$

$$
\begin{aligned}
p &= E(s|r_1^s, r_2^r, RD_2) \\
&= \frac{1}{1 + 10^{-g(RD_2)(r_1^s - r_2^r)/400}} \\
&= \frac{1}{1 + 10^{0.96404*(1504-1491)/400}} \\
&= 0.518
\end{aligned}
\tag{4.4}
$$

$$
\begin{aligned}
q &= E(s|r_2^s, r_1^r, RD_1) \\
&= 0.695
\end{aligned}
$$

Values $p = 0.518$ and $q = 0.695$ are then fed in the markov chain model from section 2.1. Calculated probability of player 1 winning a best of 3 match is $P^M = 0.011$. In this example we would most likely place our bets on the second player.

After the match is finished, we obtain the statistics. Player 1 has won 45 points out of 100 points served. This would mean that $p_{true} = 0.45$. We would use this value to update player's rankings. Refer to the example in section 3.1.

4.4 Selecting parameters

As the basis of our model is a ranking system, we need to choose the best performing one for further analysis. The parameters are optimized using the error metric presented in subsection 4.2 and Nelder Head minimization method.

To use Modified Glicko ranking system, one has to optimize its parameters: RD_{begin}, RD_{end} and RD_{step}.

We start by estimating parameters RD that are similar to those used in Glicko's examples and that stabilize rating after 20 games. Our starting parameters are therefore $RD_{begin} = 300$, $RD_{end} = 100$ and $RD_{step} = 10$. This will be our starting point for the optimization. The report for its performance can be found in appendix B.

Parameters are optimized using the error metric, presented in the subsection 4.2 and Nelder Head minimization method. We present starting points and their points of convergence in table 4.1.

Starting point			**Ending point**			**Error**
RD_{begin}	RD_{end}	RD_{step}	RD_{begin}	RD_{end}	RD_{step}	
300.0	100.0	10.0	-116.5	45.8	19.0	21171.7
90.0	45.0	15.0	55.0	45.7	19.6	21171.0
55.0	45.0	10.0	55.0	45.7	10.1	21171.0
30.0	10.0	4.0	32.5	45.8	0.17	21171.7

Table 4.1: Starting and ending points of base model parameter optimization B.

Parameters converge from most points to values $RD_{begin} = 55.0$, $RD_{end} = 45.7$ and $RD_{step} \geq 9.3$. Since RD becomes smaller then RD_{end} after one match, it doesn't matter what the value of RD_{step} is, as long as it is larger then 9.3. Model can also converge to local optimum $RD_{begin} \leq 45.8$, $RD_{end} = 45.8$. Attentive reader observes that for those values ranking system is an Elo ranking system.

Unfortunately there is only a small advantage of adding a RD variable. In the improved model, we will use the Elo ranking system instead of the modified Glicko ranking system.

4.5 Graphical tool

The presented model can be looked at as a metric for player's form. Each player is assigned two values, serve and return strengths. These can be presented as points in a 2-D plane.

We create a web application to graphically present how rankings change in time. How this application looks like, can be seen in the figure 4.1.

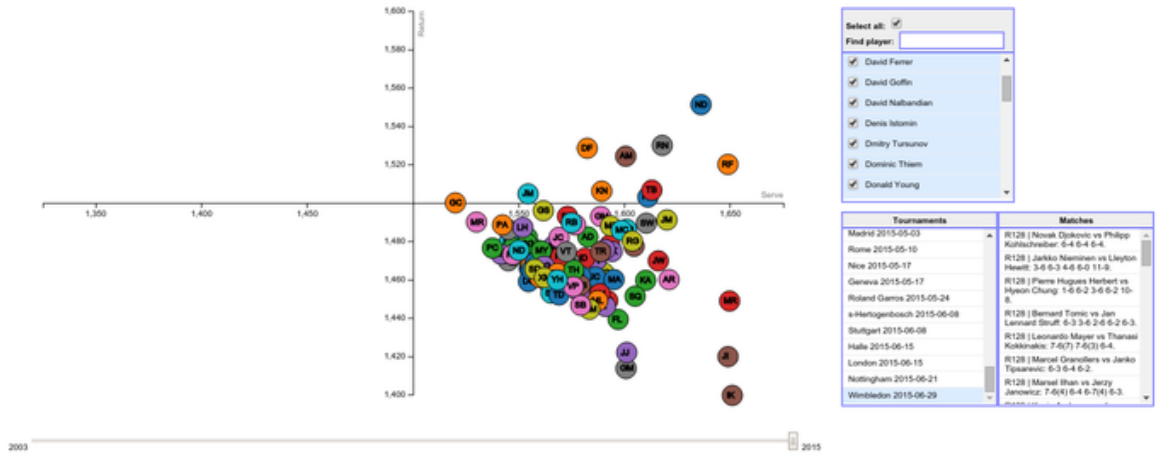

Figure 4.1: Graphical tool presenting form of the players.

The application uses data from year 2003 to year 2015. For a clearer presentation, values are selected on tournament by tournament basis and smoothed with an exponential function.

The created graphical tool is used to spot patterns and problems in the existing model. Screenshots are used to demonstrate model analysis.

Chapter 5

Problems in the base model

In this chapter we will analyse our base model and data used, mostly for purpose of finding room for improvement and to create a better, improved model later on. Each section presents an issue that we discovered while analysing our model and proposes a viable solution.

5.1 Skewed results

We observe our model's expected probabilities of players winning a point on serve (p, q) against the actual percentage of points won on serve. In the figure 5.1 we compare these probabilities.

Our model has an average prediction of 0.6087, while the actual average is 0.6312. There is an average difference of 0.0224, meaning that our model underestimates probabilities of players winning point on serve. The variance of difference between the predicted and the actual value is 0.0895.

Data is skewed for rankings as well. The figure 5.2 displays a screeshot of the graphical tool, presenting players' rankings at time of Wimbeldon 2015. We can see that the serve ratings are in general larger then the return ratings.

The obtained result seen on the figure 5.2 is expected, since the player on serve has an advantage. However, this is the reason why our base model constantly underestimates probabilities of players winning a point on serve.

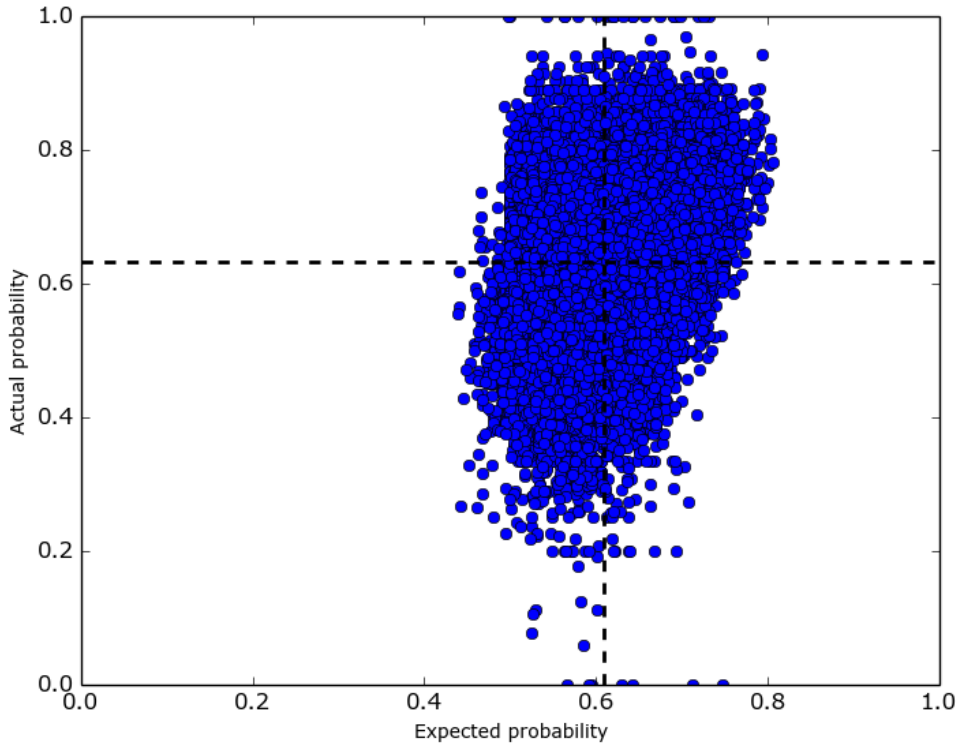

Figure 5.1: Expected vs actual probabilitites of players winning a point on serve.

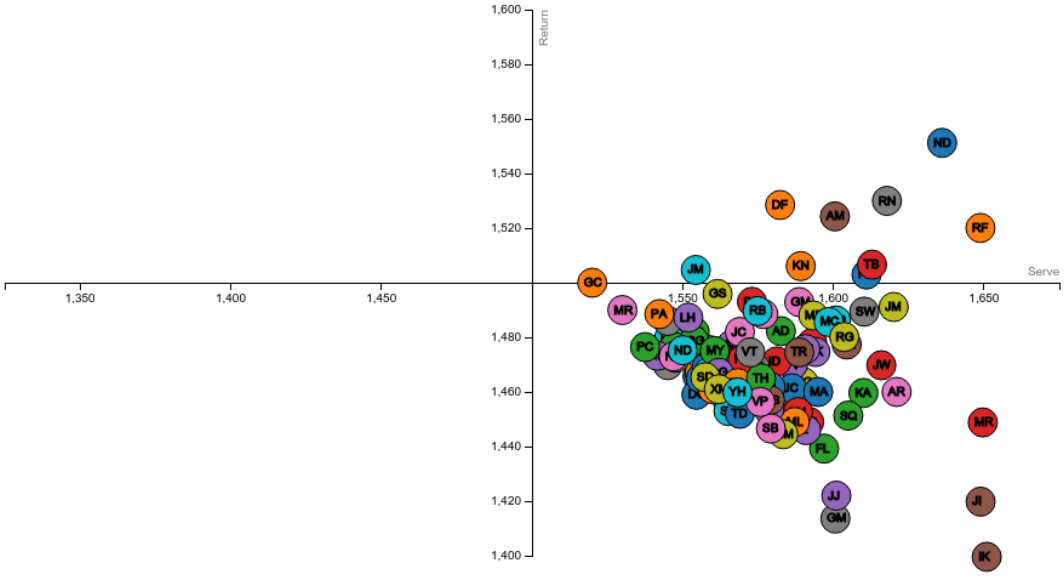

Figure 5.2: Rankings of top 100 players presented in a 2-D plane.

5.1.1 Solution

The base model seems to malfunction since two groups of ratings, serve and return ratings, have a different average.

$$\begin{aligned} p &= E(s|r_1^s, r_2^r, RD_2) \\ q &= E(s|r_2^s, r_1^r, RD_1) \end{aligned} \tag{5.1}$$

As the above equations don't return correct results, they should be modified accordingly. The solution is adding another variable that implies a built-in difference between two groups of rankings. We rewrite equations as:

$$\begin{aligned} p &= E(s|r_1^s, r_2^r, RD_2) + e \\ q &= E(s|r_2^s, r_1^r, RD_1) + e \end{aligned} \tag{5.2}$$

We add e when calculating values p and q. However, we must also substract e from the actual results when updating ratings.

For $e = 0.1$ we create a scatter plot and calculate averages. The calculated average is now 0.626, differing from the actual average for 0.005. Furthermore, the variance has been reduced to 0.0835. The new result can be seen in the figure 5.3.

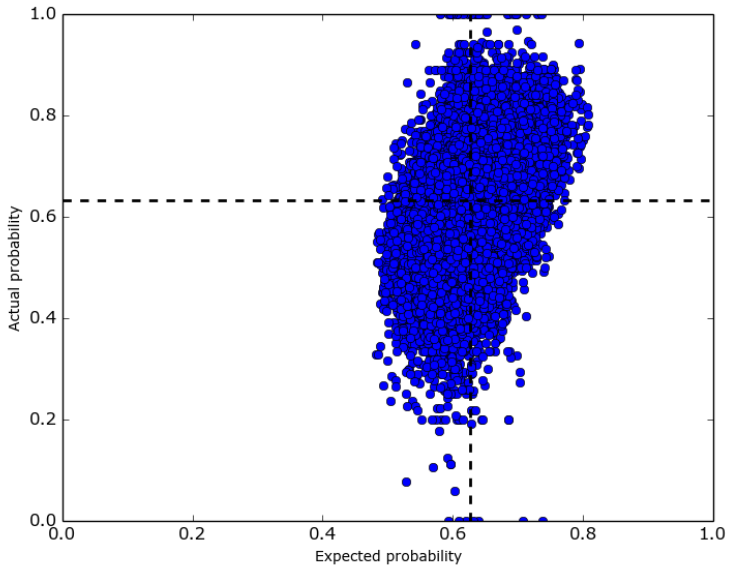

Figure 5.3: Expected vs actual probability of a player winning point on serve for adjusted equations.

5.2 Impact of the surface

So far a huge aspect of game analysis, that we have neglected, is the impact of the surface. Each surface has characteristics that affect the style of playing. It is well known that some players have a varying performance on different surfaces.

Our base model completely ignores the impact of the surface, which is a large blunder. With our approach to modelling, there is no straightforward approach to incorporate it in our model. In this section we explore this impact and how it could be used to enhance our base model.

5.2.1 Surface comparison

The most important difference between surfaces is their hardness, which influences the strength of the serve. Faster the surface, higher the percentage of players winning a point on serve. The figure 5.4 compares the probabilities of players winning a point on serve on different surfaces. Clay, as the slowest surface, has an average probability of 0.611, hard 0.637, carpet 0.653 and grass has an average of 0.657. We can see how probability increases from the slowest to the fastest surface.

Our model measures players' serve and return strengths. The intuitive guess would be that players with a higher serve rating have more of an advantage on a fast surface like grass. We confirm our assumptions with the figures 5.5 and 5.6. The serve rating shows an increasing trend from slowest to fastest surface, while return rating has a descreasing trend.

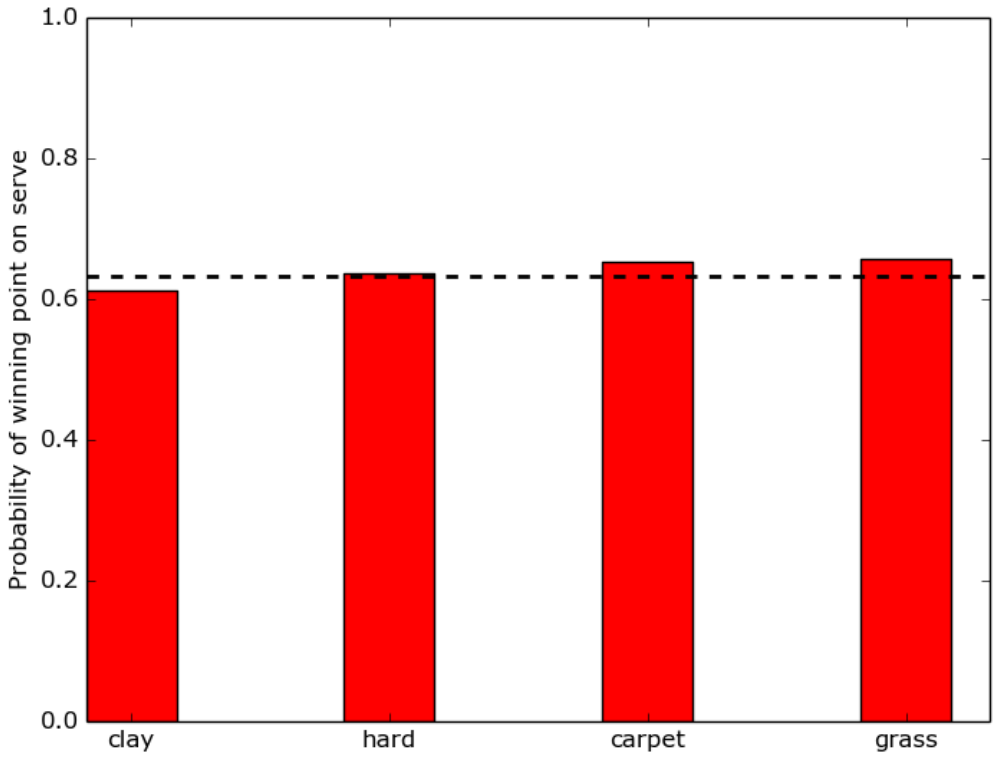

Figure 5.4: Probabilities of players winning a point on serve on different surfaces.

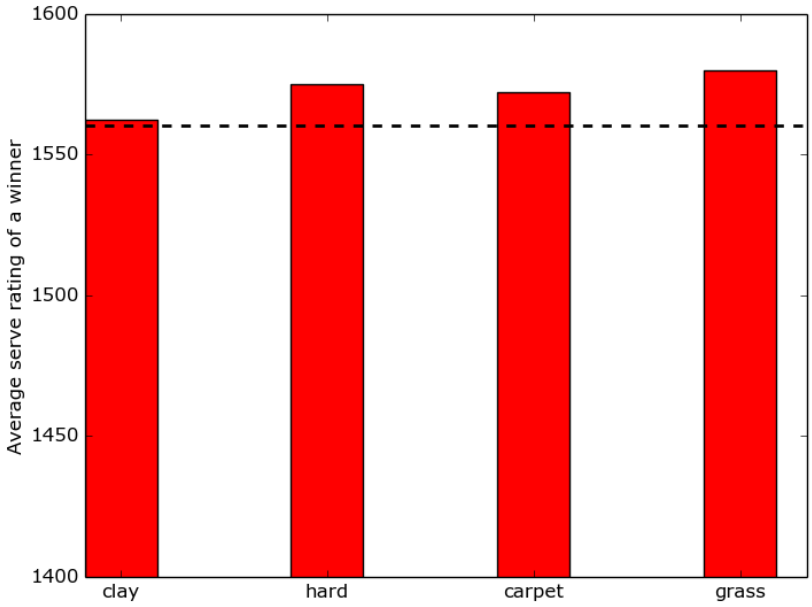

Figure 5.5: Average serve rating of a winner on different surfaces.

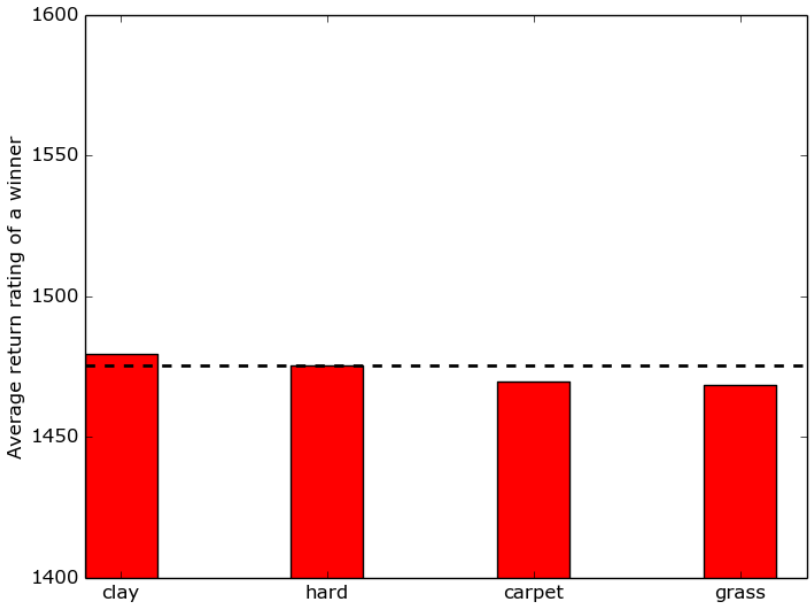

Figure 5.6: Average return rating of a winner on different surfaces.

5.2.2 Case study

We study how a surface might impact player's performance on Andy Murray. We have $n = 662$ his matches in total in our database. In the table 5.1 one can find statistics for Andy Murray playing on different surfaces.

Surface	Matches won	Points won on serve	Points won on return	n
Hard	0.7748	0.6558	0.4305	444
Clay	0.6636	0.6236	0.4320	107
Grass	0.8300	0.7010	0.4003	100
Carpet	0.7273	0.6509	0.4001	11
All	0.7644	0.6573	0.4257	662

Table 5.1: Statistics for Andy Murray on different surfaces

From the data in the table 5.1 one can tell that Andy has a superior perfromance on grass, while his weak surface is clay. Andy Murray wins 0.7010 of points on serve, when playing on grass. This is 0.0437 more then his average. One would think that this is something we should be able to incorporate in our model.

Lets continue observing Murray's performance on grass. We model his points won on serve as a normal distribution, as seen on the figure 5.7. While normal distribution fits data well, first standard deviation interval is suprisingly wide and even contains overall average. According to normal distribution there is less then 68% probability that Andy Murray's average serve performance on grass is actually better than his average serve performance. This is rather surprising given extensive data we have available for Andy Murray.

Lets try to model Andy's points won on serve as a Bernoulli distribution instead. First standard deviation interval is now way tighter as expected on such a data set.

A very straightforward idea to adding the surface impact to our model, is to simply adjust player's serve or return strength where it is statistically significant. However, one should not ignore the built-in differences between the surfaces that are not player specific. We try to bring our observations together in a coherent model in the next subsection.

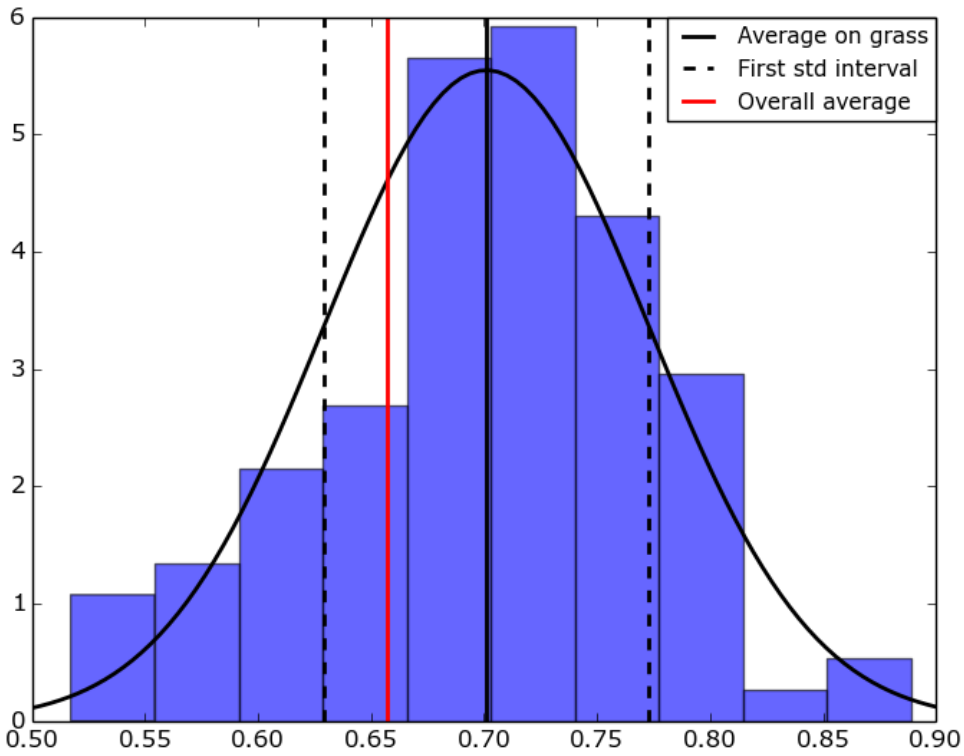

Figure 5.7: Probability distribution of Andy Murray winning a point on serve on grassy surface. We model data with a normal distribution.

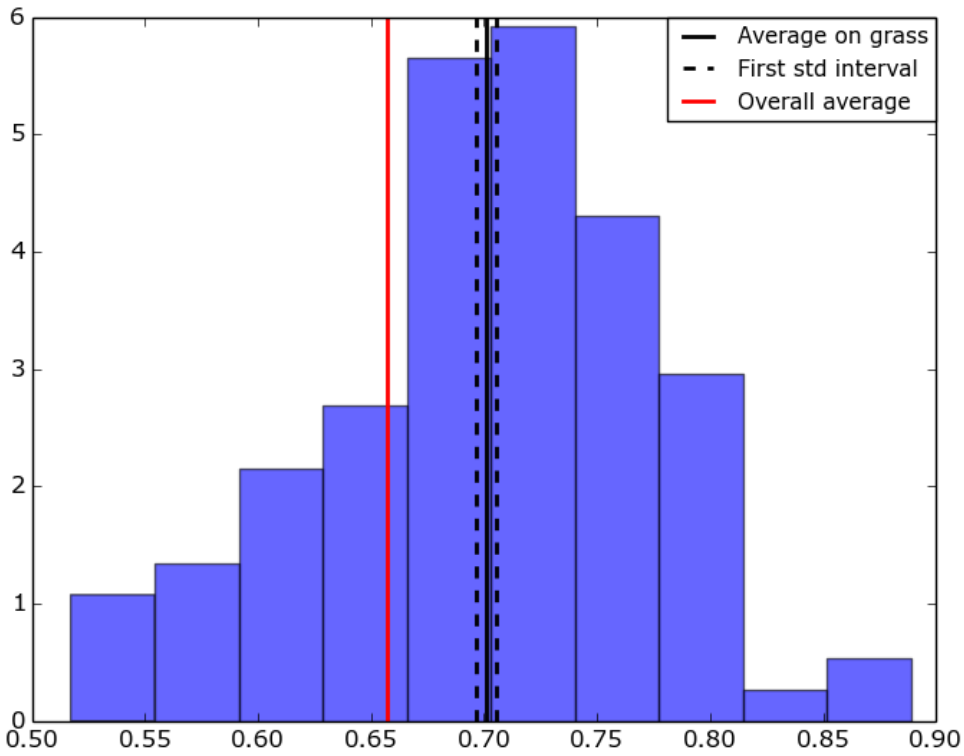

Figure 5.8: Probability distribution of Andy Murray winning a point on serve on grassy surface. We model data with a Bernoulli distribution.

5.2.3 Solution

Building on top of our observations, we try to formulate the surface impact. We will modify the probabilities of players winning a point on serve (p, q), with equation:

$$p' = p + p_a \tag{5.3}$$

To calculate p_a, we will use a procedure presented later in the subsection. First, we calculate the input variables $input$ and h. The only variable that we have to adjust is α, which we set between 0 and 1. A higher value means that we are more careful when including the impact of the surface.

$$\begin{aligned} s_1 &= s_1^{surface} - s_1^{all} \\ r_2 &= r_2^{surface} - r_2^{all} \\ input &= s_1 - r_2 \end{aligned}$$

$$\begin{aligned} h_1 &= \sqrt{\frac{s_1^{surface}(1 - s_1^{surface})}{100 * n_1^{surface}}} * z_{1-\alpha/2} \\ h_2 &= \sqrt{\frac{r_2^{surface}(1 - r_2^{surface})}{100 * n_2^{surface}}} * z_{1-\alpha/2} \\ h &= h_1 + h_2 \end{aligned} \tag{5.4}$$

- $s_1^{surface}$ - average percentage of points player 1 won on serve, on a specific surface
- s_1^{all} - average percentage of points player 1 won on serve
- $r_2^{surface}$ - average percentage of points player 2 won on return, on a specific surface
- r_2^{all} - average percentage of points player 2 won on return
- $n_1^{surface}$ - number of matches player 1 played on a specific surface
- $n_2^{surface}$ - number of matches player 2 played on a specific surface
- α - confidence coeficient

We get value p_a using the next algorithm:

Algorithm 1 Calculate surface impact.

1: **procedure** CALCULATE p_a
2: if $input + h < 0 : p_a = input + h$
3: elif $input - h > 0 : p_a = input - h$
4: else $p_a = 0$
5: return p_a
6: **end procedure**

5.3 Momentum

The last analysed area, in which our base model is lacking, is the lack of the momentum. There is no variable in the model that keeps track of volatility of player's performance. If a player suddenly starts improving very fast, the system will take a while to catch up. This problem can also be looked at as a player's form prediction.

In this subsection we analyse how our system underestimates Kei Nishikori's performance and suggest possible improvements of model that would track player's momentum.

5.3.1 Case study

We show the lack of momentum of our model on an example. Kei Nishikori had a very steady climb from 2011 to 2015. When he participated in Wimbeldon 2015, he was ranked 4th in ATP rankings, while our model ranks him 10th (if we rank players according to sum of the serve and return ratings).

The figure 5.9 shows a screenshot from graphical tool of Kei Nishikori's progress. The circle is set in the middle of the year 2012. We can see steady climb and a jump in 2015. The figure 5.10 compares Kei Nishikori's ATP ranking to his base model ranking. Our model is shadowing ATP ranking, but with a delay, since Elo ranking system takes a while to converge to the actual ranking.

5.3.2 Solution

Two good solutions exist for our problem at hand. However, neither of them could be successfully integrated in our model.

Firstly, Glicko's ranking system Glicko2 ([14]) should be exactly what we want. It includes a variable that keeps track of player's ranking volatility. Should player start improving faster, his rating would improve by even larger margin, until it stabilizes. We had no success with tuning Glicko2 ranking system to fit our model. The report

for best attempt can be found in appendix B.3.

Secondly, some research was done for adding momentum to Elo ranking system by Bester [10]. He presents three different approaches for adding momentum to Elo ranking system. None of them improved the performance of our model.

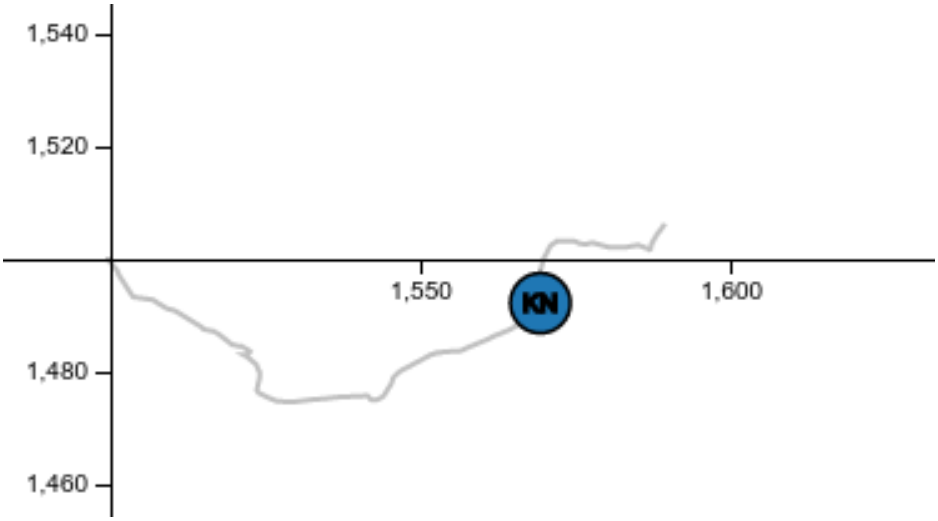

Figure 5.9: Screenshot from the graphical tool of Kei Nishikori progress.

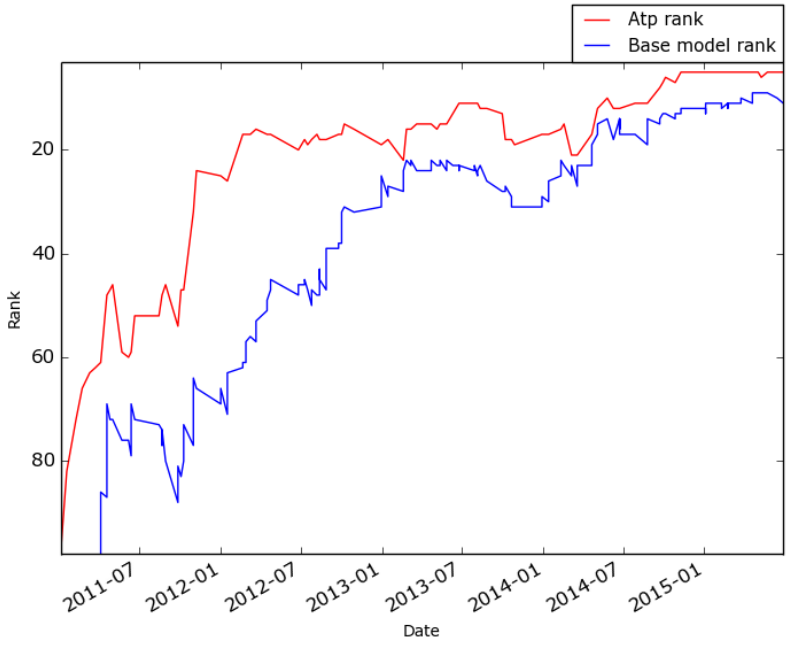

Figure 5.10: Comparison of Kei Nishikori's atp rank and base model rank.

Chapter 6

Improved model

Using the observations from the previous section, we try to improve our base model. Instead of modified Glicko ranking system, which gave only a minor advantage, we use Elo ranking instead. We sacrifice minor error margin for simplicity. Moreover, we add the edge constant as discussed in the subsection 5.1. Finally, we add the surface impact parameters.

6.1 Model

With r_1^s and r_1^r we denote the serve and return ratings of player 1. Similarly we use r_2^s and r_2^r for ratings of player 2. Lets recall Elo ranking equation for expected score from the subsection 2.2.1:

$$E(r_1, r_2) = \frac{1}{1 + 10^{(r_1 - r_2)/400}} \tag{6.1}$$

We calculate probabilities of players winning a point on serve using the next two equations:

$$\begin{aligned} p &= E(r_1^s, r_2^r) + 0.1 + p_a \\ q &= E(r_2^s, r_1^r) + 0.1 + q_a \end{aligned} \tag{6.2}$$

Where p_a and q_a are surface adjustments and will be set to $p_a = 0$ and $q_a = 0$ until later.

After each match we update the ratings of both players. Lets use s_1 to denote the percentage of points on serve player 1 has won and s_2 for player 2.

We update the ratings using the next equations:

$$\begin{aligned} r_1^s &= K * (s_1 - E(r_1^s, r_2^r)) \\ r_1^r &= K * (E(r_2^s, r_1^r) - s_2) \\ r_2^s &= K * (s_2 - E(r_2^s, r_1^r)) \\ r_2^r &= K * (E(r_1^s, r_2^r) - s_1) \end{aligned} \tag{6.3}$$

As surface impacts p_a and q_a are set to 0 for now, the only parameter we optimize is parameter K. We will show how we optimized it in the next section.

6.2 Selecting parameters

As there is only one parameter to be optimized, we can draw a figure of the model error as a function of parameter K, shown on figure 6.1.

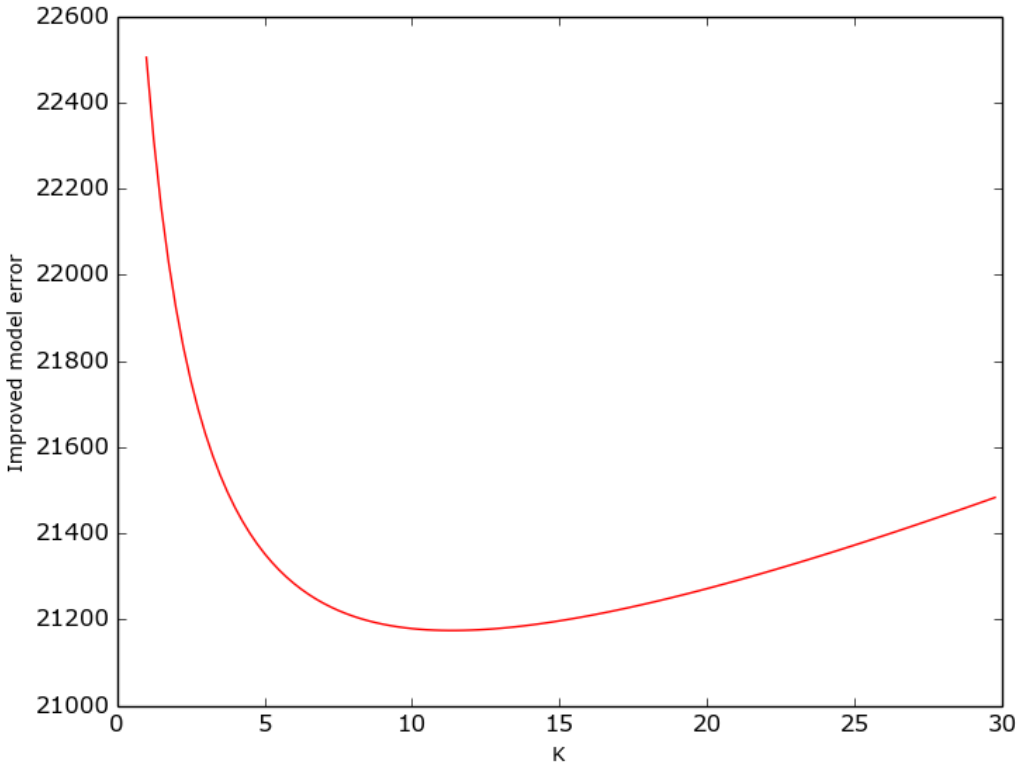

Figure 6.1: Error of the improved model for different values K.

The function of parameter K is convex with minimum around 12. For most optimization methods and starting points, we should obtain the optimal value. We run optimization twice, starting from $K = 6$ once and once from $K = 20$. In both cases

parameter converges to $K = 11.35$ with an error of 21174.8. Model's error is slightly larger then the one in the model using the modified Glicko ranking system.

Reports for improved model at values $K = 6$, $K = 20$ and $K = 11.35$ can be found in appendix B. For all three parameters, our model has above 66% prediction percentage and beats betting exchanges over 1500 bets placed, without any betting strategy.

6.3 Adding surface impact

In the section 5.2 we have analysed the impact of the surface and suggested equations that could be used to improve the existing model. Surface impact has just one parameter, that is α. When adding the surface impact to our improved model, we will calculate p and q using the next two equations:

$$\begin{aligned} p &= E(r_1^s, r_2^r) + 0.1 + p_a \\ q &= E(r_2^s, r_1^r) + 0.1 + q_a \end{aligned} \tag{6.4}$$

Where p_a and q_a are calculated as in the algorithm 1.

We test our model for interesting values α. The results are presented in table 6.1. As expected, for $\alpha = 1.0$ model performs as surface impact wouldn't be taken into consideration. On the other spectrum, for alpha smaller than 0.5 our model performs worse than without surface impact addition. This is because model takes into cosideration too many unreliable adjustments.

α	Improved model error
0.25	21401.2
0.5	21261.3
0.68	21172.6
0.85	21091.7
0.95	21051.0
0.975	21043.1
0.9875	21043.9
0.0925	21074.0
1.0	21174.8

Table 6.1: Improved model with surface impact, error for different values of α.

Chapter 7

Applications

In this chapter we present three main areas in which our model could be used. We mostly provide guidelines for application and possible modifications of our model for anyone who would want to do more specialised research in areas presented. For each area we demonstrate application on an example.

7.1 Player training

Our model in combination with a graphical tool that we provide could be used to guide a training of a player. Should player improve in return ranking but not in serve ranking, we could adjust player's training to include more serve training and playing on serve. Opposite holds for player whose serve ranking is stronger than return ranking.

On the figure 7.1 we can observe the ranking of Ivo Karlovic. He has a superior serve ranking, as strong as Novak Djokovic's, nonetheless his return ranking is rather poor. While we don't doubt that his training was specialized to correct this, similar ranking approach used in our model could provide an insight in finer details.

Parameters of our model were trained to minimize the error of our metric, however this could be adjusted depending on the application we use our model for. When training a player, one could set parameter K to be very high and observe the rating changes in a small timeframe according to changed training practices.

We realise that the current state of the model will not provide any actually useful advice on training players, however we believe that ranking approach of different stats might be the way to go to spot weaknesses of players. Using software might be very useful in junior categories where players get less attention.

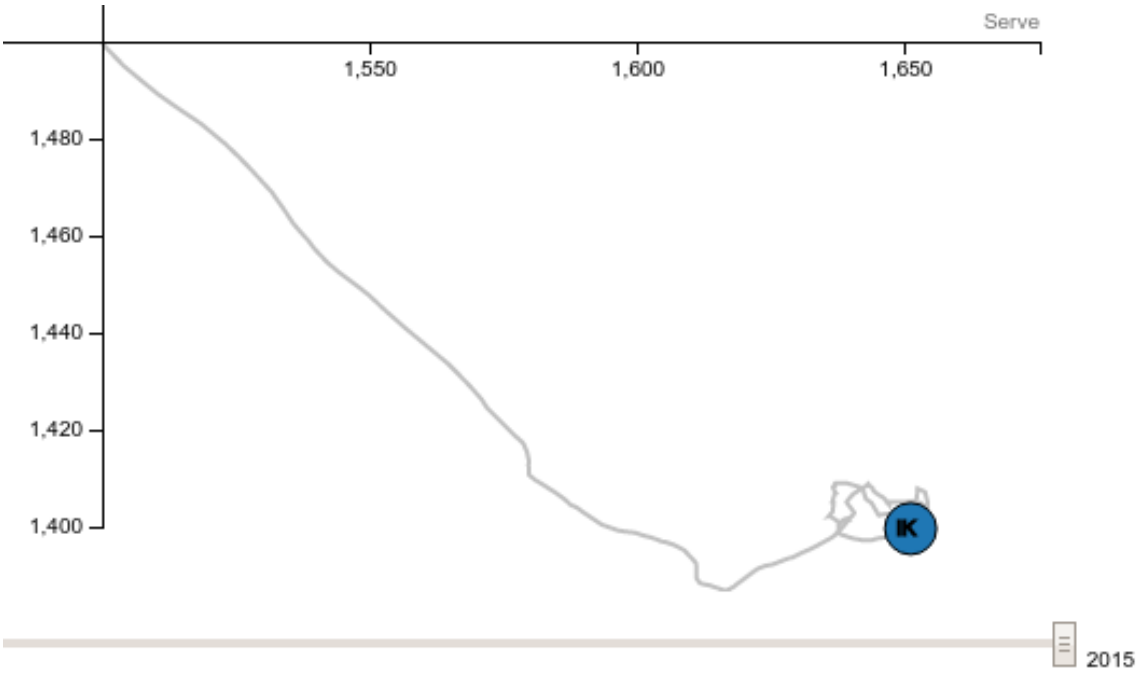

Figure 7.1: Screenshot from the graphical tool showing Ivo Karlovic's progress.

7.2 Match duration

As mentioned, media has a great interest in estimating the duration of a tennis match. Since it is not fixed, media schedule has to be estimated. In this section we present a simple way of estimating time duration using Monte Carlo simulation and normal distribution fitting.

We start by comparing the number of points played with the length of a match in minutes. The figure 7.2 demonstrates that there exists a linear relationship. If we denote the length in minutes as m and points played as p, we can write the relationship with the next linear equation:

$$m = 0.6756 * p - 1.257 \tag{7.1}$$

Using Monte Carlo simulation we can find the distribution of points played, convert it to match duration using the above equation and then fit the normal distribution. So obtained normal distribution can be used to obtain the expected length of the match, but more importantly also the confidence intervals.

We test our approach on Wimbeldon 2015 finals, where Novak Djokovic defeated Roger Federer after 2 hours and 56 minutes of intense tennis. Our model estimates probability p of Novak Djokovic winning a point on serve against Roger Federer at an interesting number of $p = 0.666$. Probability of Roger Federer winning a point on

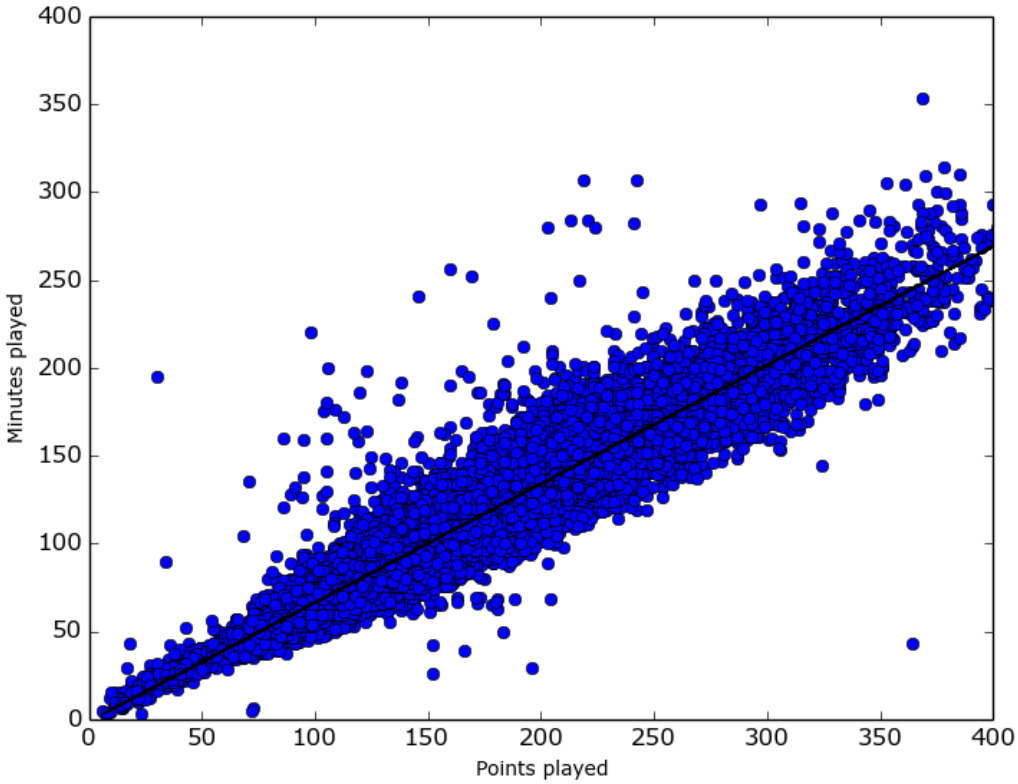

Figure 7.2: Correlation between the number of points played and minutes played.

serve is estimated at $q = 0.640$.

Values p and q are fed in Monte Carlo simulation model implemented in Cython to obtain the distribution of points played. We use the equation 7.1 to convert these values to minutes played and finally fit the normal distribution to data, so that we can obtain the distribution presented in the figure 7.3.

Our model estimates the duration at 179 minutes, a precise estimate that was off by only 7 minutes.

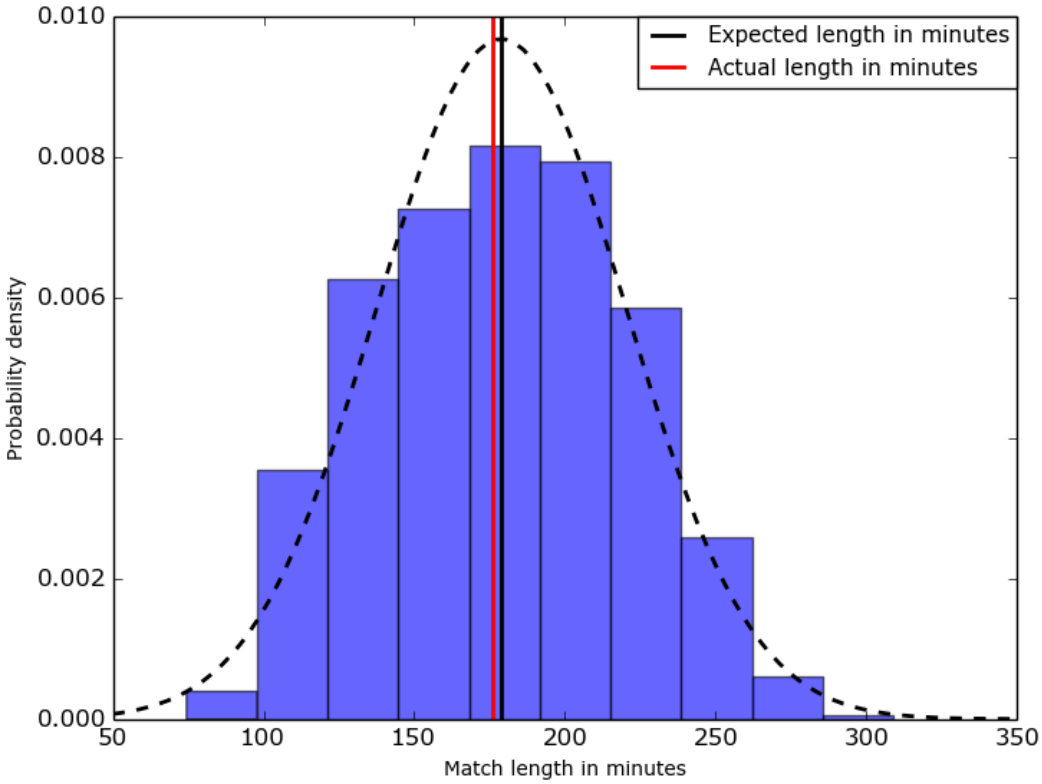

Figure 7.3: Generated distribution of match length in minutes for Wimbeldon 2015 finals.

7.3 Betting

A very popular area of application for tennis match prediction is betting. Both bettors and betting exchanges have a great interest in predicting the outcome of tennis matches.

To successfully bet against a betting exchange more then just good predictions are neccessary. The reader that read reports from appendix B observed crazy variance that is involved in betting. Should one want to use a model to win against a betting exchange, good bankroll management and strong betting strategy must be employed.

To cross validate our model and draw a comparison to common opponent model ([16]), we test how we could use our model to bet in 2012. Common opponent model is put to the test on 2173 matches played in 2012. With 1228 bets placed, common opponent model returns 3.80% on each money unit invested.

Our data for 2012 contains 3110 matches played and uses odds of a major betting exchange. We have placed bets on 2614 of them. We didn't use any sophisticated

betting strategy - we bet 1 money unit whenever we estimated that odds are in our favor. In total, our model generated 2.5% return on each invested unit. Commulative profit is presented in the figure 7.4.

Looking at the results and considering the possibility of reinvesting the money, one might be mislead that we have just discovered a gold mine waiting for us. It must be noted that betting exploits very small margins that are often high variance and a sample of 3000 matches is rather small. Nonetheless, we believe that with a proper betting strategy our model could be used to successfully bet against the exchange or be used to enhance an existing model used for betting.

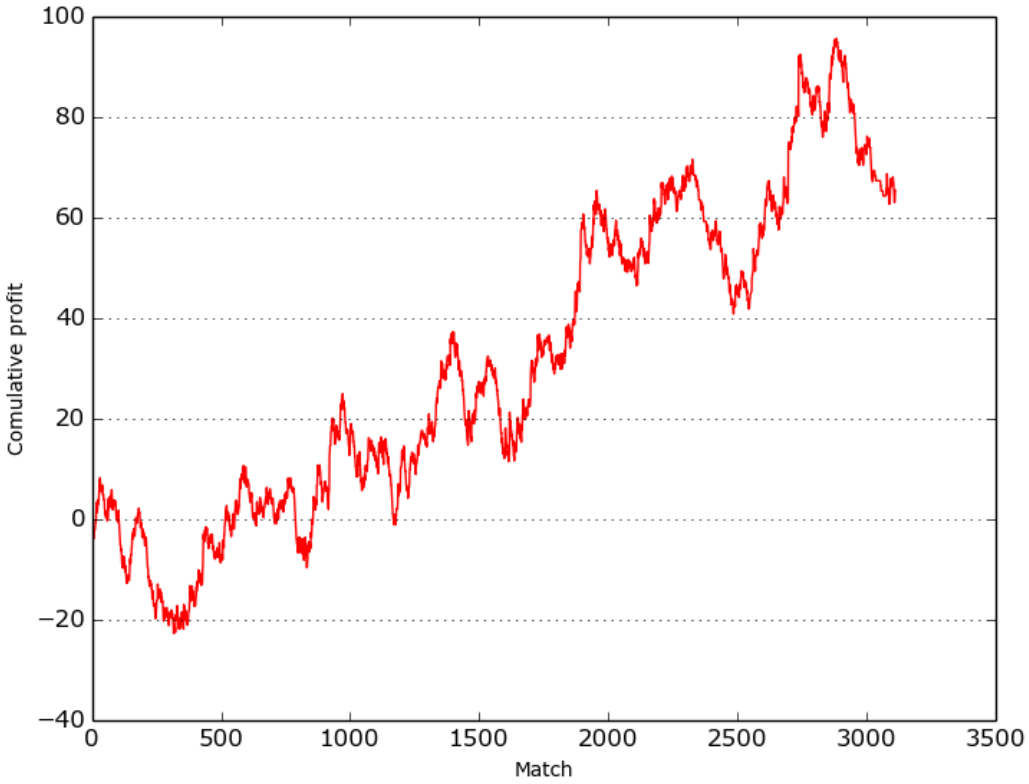

Figure 7.4: Commulative profit when betting against a betting exchange in 2012.

Chapter 8

Conclusion

To summarize, we look at what this project achieved and what are the possible directions for the future development. We start this chapter by summarizing the achievements and then provide guidelines for anyone that might be interested in further development.

8.1 Achievements

This project deals with tennis match prediction. Combining the techniques used in the existing, publicly available models we designed a new model.

To implement our model, we created our own development framework. This framework is very lightweight and is very easy to start implementing in. It will be released to public and anyone that wants to pick up the project should have no problems getting up to speed.

While our model is based on existing models, it takes an innovative approach to modelling. In this project we took a completely new approach to estimating probabilities of players winning a point on serve.

We attempted to develop a new ranking system that would suit the needs of our model the best. Unfortunately we were only able to get a minor improvement in model's error.

The output of our model, players' ratings, is a great metric for other types of analysis. We created a graphical tool that provides insight in how players perform in time.

We applied our model to the match length prediction and provide a reasonable method to get a good estimate of the match length.

Finally, our model has beaten betting exchanges using different parameters and data sets. We are convinced that our model could be used to defeat betting exchanges.

8.2 Future work

There are several directions in which this project might be continued. In this section we will give ideas for anyone that might be interested to build on our work. As our work provides a framework, with ready to use database, ranking systems and models, one can start with development easily.

Firstly, we believe that adding momentum to Elo ranking system has a very high potential. One could try to combine Elo and Glicko2 ranking systems to create a specific ranking system. Another possibility is to somehow quantify player's momentum and modify parameter K of the Elo ranking, based on that momentum.

To continue, we suggest a different approach to measure the surface impact. In this thesis we tried to measure the surface impact by averaging each player's statistics. One could, instead of this approach, create additional players' rankings for each of the surfaces. Surface specific ranking could be averaged with general ranking to obtain the probabilities.

We proposed three areas of application; player training, match length prediction and betting. Application of our model in each of these areas could be researched in depth individually.

The ranking system could be used to rank more then just players' serve and return ratings. One could break this down to first serve, second serve, play after first serve, etc. This should provide more insight in specific strengths and weaknesses of each player.

Lastly, one could use our model, together with a betting strategy to defeat a betting exchange. A variable could be added to Elo ranking system to track reliability of players' rankings, as was done in the modified Glicko ranking system. Rating reliability is then used as a part of the betting strategy.

Bibliography

[1] Tennis data. Tennis data source. http://www.tennis-data.co.uk/alldata.php, 2015 (visited on 03/09/2015); pages 16

[2] Wikipedia. Chess rating systems. https://en.wikipedia.org/wiki/Chess_rating_system, 2015 (visited on 03/09/2015); pages 10

[3] Wikipedia. Elo rating system. https://en.wikipedia.org/wiki/Elo_rating_system, 2015 (visited on 03/09/2015); pages 10

[4] Wikipedia. Tennis scoring system. https://en.wikipedia.org/wiki/Tennis_scoring_system, 2015 (visited on 03/09/2015); pages 5

[5] Jeff Sackmann. Tennis data source. https://github.com/JeffSackmann/tennis_atp, 2015 (visited on 03/09/2015); pages 16

[6] Tennis game markov model. http://i42.tinypic.com/5evfa9.jpg, 2015 (visited on 03/09/2015); pages 1, 6

[7] Tristan Barnett. Mathematical modelling in hierarchical games with specific reference to tennis. PhD dissertation, Swinburne university, 2006; pages 2, 5, 18

[8] Tristan Barnett and Stephen R. Clarke. Using Microsoft Excel to model a tennis match. *6th Conference on Mathematics and Computers in Sport (G. Cohen ed.)*. pp. 63–68, 2002; pages 2, 5

[9] Tristan Barnett and Stephen R. Clarke. Combining player statistics to predict outcomes of tennis matches. *IMA Journal of Management Mathematics*, vol. 16, no. 2, pp. 113–120, 2005; pages 9, 18

[10] D. W. Bester. Introducing Momentum to the Elo rating System. http://natagri.ufs.ac.za/dl/userfiles/Documents/00002/2069_eng.pdf, (visited on 03/09/2015); pages 33

[11] M. E. Glickman. A Comprehensive Guide to Chess Ratings. *American Chess Journal*, vol. 3, pp. 59–102, 1995; pages 10

[12] M. E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics*, vol. 48, no. 3, pp. 377–394, 1999; pages 16

[13] M. E. Glickman. Glicko rating system. `http://www.glicko.net/glicko/glicko.pdf`, 1995 (visited on 03/09/2015); pages 10, 11, 13

[14] M. E. Glickman. Glicko 2 rating system. `http://www.glicko.net/glicko/glicko2.pdf`, 2001 (visited on 03/09/2015); pages 10, 12, 13, 32

[15] R. Herbrich, T. Minka, and T. Graepel. TrueSkillTM: A Bayesian Skill Rating System. *Advances in Neural Information Processing Systems (eds. B. Scholkopf, J. Platt, and T. Hoffman)*, vol. 19, pp. 569–576, 2006; pages 10, 12

[16] William J. Knottenbelt, Demetris Spanias, and Agnieszka M. Madurska. A common-opponent stochastic model for predicting the outcome of professional tennis matches. *Computers and Mathematics with Applications*, vol. 64, no. 12, pp. 3820–3827, 2012; pages 9, 40

[17] Agnieszka M. Madurska. A Set-By-Set Analysis Method for Predicting the Outcome of Professional Singles Tennis matches. Bachelor's thesis, Imperial College London, 2012; pages 2, 5

[18] Ian McHale and Alex Morton. A Bradley-Terry type model for forecasting tennis match results. *International Journal of Forecasting*, vol. 27, no. 2, pp. 619–630, 2011; pages 2, 16

[19] A. J. OMalley. Probability formulas and statistical analysis in tennis. *Journal of Quantitative Analysis in Sports*, vol. 4, no. 2, p. 15, 2008; pages 8

Appendices

Appendix A

Tables used in O'Malleys equations

A.1 Table B

B	1	2	3	4	5	6
1	1	3	0	3	0	0
2	3	3	1	3	0	0
3	3	4	0	2	1	0
4	6	2	2	4	0	0
5	12	3	1	3	1	0
6	3	4	0	2	2	0
7	4	2	3	4	0	0
8	24	3	2	3	1	0
9	24	4	1	2	2	0
10	4	5	0	1	3	0
11	5	1	4	5	0	0
12	40	2	3	4	1	0
13	60	3	2	3	2	0
14	20	4	1	2	3	0
15	1	5	0	1	4	1
16	1	0	5	5	0	1
17	25	1	4	4	1	1
18	100	2	3	3	2	1
19	100	3	2	2	3	1
20	25	4	1	1	4	1
21	1	5	0	0	5	1

Table A.1: Table B used in O'Malleys equations.

A.2 Table A

A	1	2	3	4	5	6
1	1	3	0	4	0	0
2	3	3	1	4	0	0
3	4	4	0	3	1	0
4	6	3	2	4	0	0
5	16	4	1	3	1	0
6	6	5	0	2	2	0
7	10	2	3	5	0	0
8	40	3	2	4	1	0
9	30	4	1	3	2	0
10	4	5	0	2	3	0
11	5	1	4	6	0	0
12	50	2	3	5	1	0
13	100	3	2	4	2	0
14	50	4	1	3	3	0
15	5	5	0	2	4	0
16	1	1	5	6	0	0
17	30	2	4	5	1	0
18	150	3	3	4	2	0
19	200	4	2	3	3	0
20	75	5	1	2	4	0
21	6	6	0	1	5	0
22	1	0	6	6	0	1
23	36	1	5	5	1	1
24	225	2	4	4	2	1
25	400	3	3	3	3	1
26	225	4	2	2	4	1
27	36	5	1	1	5	1
28	1	6	0	0	6	1

Table A.2: Table A used in O'Malleys equations.

Appendix B

Reports

B.1 Example

Model name: Random
Description: Report of a random model used as an example.

Error: 35101.25224
Bias: 0.50307
Sqme: 0.33622

Correct predict percentage:

2012	2013	2014	2015
0.5047	0.49043	0.48734	0.50732

Predict mean: 0.49745
Predict std: 0.00868
Predict 95% confidence interval: (0.46983, 0.52507)

Bets placed:

2012	2013	2014	2015
443	452	410	224

Betting ROI:

2012	2013	2014	2015
−0.04619	0.04049	−0.14341	−0.22254

ROI mean: -0.07247
ROI std: 0.09285
ROI 95% confidence interval: (-0.36796, 0.22302)

B.2 Barnett ad hoc

Model name: Barnett
Description: Ad hoc implemetnation of Barnett's averaging statistics model.
Parameters:

Error: 29949.71382
Bias: 0.48248
Sqme: 0.29597

Correct predict percentage:

2012	2013	2014	2015
0.55134	0.52412	0.52109	0.52439

Predict mean: 0.53024
Predict std: 0.01225
Predict 95% confidence interval: (0.49124, 0.56923)

Bets placed:

2012	2013	2014	2015
446	446	412	222

Betting ROI:

2012	2013	2014	2015
−0.13702	−0.00253	−0.12274	−0.13036

ROI mean: -0.09289
ROI std: 0.05831
ROI 95% confidence interval: (-0.27846, 0.09268)

B.3 Double glicko2 model

Model name: DoubleGlicko2
Description: Base model using Glicko2 ranking system instead of modified Glicko ranking. Parameters are set to defaults suggested by Glicko.
Parameters:

$$tau : 1, phi : 350.0, sigma : 0.06, eps : 1e-06$$

Error: 32567.94229
Bias: 0.38071
Sqme: 0.26045

Correct predict percentage:

2012	2013	2014	2015
0.68185	0.64663	0.65127	0.64512

Predict mean: 0.65622
Predict std: 0.01497
Predict 95% confidence interval: (0.60857, 0.70386)

Bets placed:

2012	2013	2014	2015
445	456	416	222

Betting ROI:

2012	2013	2014	2015
0.06858	−0.1025	0.02788	0.03707

ROI mean: 0.00235
ROI std: 0.06983
ROI 95% confidence interval: (-0.21988, 0.22457)

B.4 Base model 1

Model name: DoubleModifiedGlicko
Description: Base model with parameters set at a starting point of our optimization.
Parameters:

$$RD_{step} : 10.0, RD_{begin} : 300.0, RD_{end} : 100.0$$

Error: 22672.95224
Bias: 0.37766
Sqme: 0.21810

Correct predict percentage:

2012	2013	2014	2015
0.69993	0.6719	0.67216	0.67683

Predict mean: 0.68020
Predict std: 0.01156
Predict 95% confidence interval: (0.64343, 0.71698)

Bets placed:

2012	2013	2014	2015
437	445	409	218

Betting ROI:

2012	2013	2014	2015
0.05295	−0.0189	0.03421	−0.06743

ROI mean: 0.00929
ROI std: 0.04249
ROI 95% confidence interval: (-0.12592, 0.14450)

B.5 Base model 2

Model name: DoubleModifiedGlicko
Description: Base model with optimized parameters.
Parameters:

$$RD_{step} : 9.3, RD_{begin} : 55.0, RD_{end} : 45.7$$

Error: 21171.03847
Bias: 0.40178
Sqme: 0.21011

Correct predict percentage:

2012	2013	2014	2015
0.6945	0.67343	0.68019	0.67866

Predict mean: 0.68170
Predict std: 0.00781
Predict 95% confidence interval: (0.65685, 0.70655)

Bets placed:

2012	2013	2014	2015
435	446	408	223

Betting ROI:

2012	2013	2014	2015
0.04011	−0.01659	0.04691	−0.01552

ROI mean: 0.01702
ROI std: 0.02973
ROI 95% confidence interval: (-0.07760, 0.11163)

B.6 Improved model 1

Model name: DoubleElo
Description: Improved model with parameter K set to 20.0.
Parameters:

$$K : 20.0$$

Error: 21272.15945
Bias: 0.38951
Sqme: 0.20970

Correct predict percentage:

2012	2013	2014	2015
0.69957	0.67496	0.68059	0.68354

Predict mean: 0.68466
Predict std: 0.00914
Predict 95% confidence interval: (0.65558, 0.71375)

Bets placed:

2012	2013	2014	2015
432	443	400	221

Betting ROI:

2012	2013	2014	2015
0.045	−0.06178	0.06337	0.02692

ROI mean: 0.01562
ROI std: 0.05150
ROI 95% confidence interval: (-0.14827, 0.17951)

B.7 Improved model 2

Model name: DoubleElo
Description: Improved model with parameter K set to 6.0.
Parameters:

$$K : 6.0$$

Error: 21283.93730
Bias: 0.41865
Sqme: 0.21260

Correct predict percentage:

2012	2013	2014	2015
0.68764	0.67075	0.67135	0.67561

Predict mean: 0.67634
Predict std: 0.00679
Predict 95% confidence interval: (0.65474, 0.69794)

Bets placed:

2012	2013	2014	2015
429	436	400	221

Betting ROI:

2012	2013	2014	2015
0.09149	−0.0286	−0.02617	−0.06516

ROI mean: 0.00129
ROI std: 0.05891
ROI 95% confidence interval: (-0.18620, 0.18877)

B.8 Improved model 3

Model name: DoubleElo
Description: Improved model with parameter K set to 11.35.
Parameters:

$$K : 11.35$$

Error: 21174.81393
Bias: 0.40136
Sqme: 0.21014

Correct predict percentage:

2012	2013	2014	2015
0.69487	0.67496	0.68019	0.68049

Predict mean: 0.68263
Predict std: 0.00740
Predict 95% confidence interval: (0.65908, 0.70618)

Bets placed:

2012	2013	2014	2015
433	448	402	220

Betting ROI:

2012	2013	2014	2015
0.05614	−0.01192	0.04114	−0.00395

ROI mean: 0.02305
ROI std: 0.02957
ROI 95% confidence interval: (-0.07105, 0.11715)

B.9 Improved model with surface impact 1

Model name: DoubleEloSurface
Description: Improved model with added surface impact. Influence of surface is set at one standard deviation.
Parameters:

$$K : 11.35, \alpha : 0.68$$

Error: 21172.64013
Bias: 0.39789
Sqme: 0.20952

Correct predict percentage:

2012	2013	2014	2015
0.69053	0.6853	0.68582	0.69146

Predict mean: 0.68828
Predict std: 0.00274
Predict 95% confidence interval: (0.67954, 0.69701)

Bets placed:

2012	2013	2014	2015
427	443	400	219

Betting ROI:

2012	2013	2014	2015
0.04145	−0.03774	0.0218	−0.00132

ROI mean: 0.00632
ROI std: 0.03169
ROI 95% confidence interval: (-0.09455, 0.10719)

B.10 Improved model with surface impact 2

Model name: DoubleEloSurface
Description: Improved model with added surface impact. Influence of surface is set at two standard deviations.
Parameters:

$$K : 11.35, \alpha : 0.95$$

Error: 21050.95251
Bias: 0.39865
Sqme: 0.20852

Correct predict percentage:

2012	2013	2014	2015
0.69487	0.68453	0.68301	0.69207

Predict mean: 0.68862
Predict std: 0.00498
Predict 95% confidence interval: (0.67277, 0.70447)

Bets placed:

2012	2013	2014	2015
434	449	401	220

Betting ROI:

2012	2013	2014	2015
0.043	−0.049	−0.01002	−0.02405

ROI mean: -0.00841
ROI std: 0.03593
ROI 95% confidence interval: (-0.12275, 0.10593)

B.11 Improved model with surface impact 3

Model name: DoubleEloSurface
Description: Improved model with added surface impact. Influence of surface is set at three standard deviations.
Parameters:

$$K : 11.35, \alpha : 0.975$$

Error: 21043.60624
Bias: 0.39916
Sqme: 0.20856

Correct predict percentage:

2012	2013	2014	2015
0.69704	0.68377	0.68059	0.69146

Predict mean: 0.68822
Predict std: 0.00645
Predict 95% confidence interval: (0.66770, 0.70873)

Bets placed:

2012	2013	2014	2015
435	447	398	221

Betting ROI:

2012	2013	2014	2015
0.05207	−0.01839	−0.00437	0.00905

ROI mean: 0.00979
ROI std: 0.02842
ROI 95% confidence interval: (-0.08066, 0.10024)

Appendix C

Guide to our framework

To continue with development of ranking models one can use the same framework as used in this thesis. In this section we will present how to set the framework, set the database and use the modules.

Setup

To obtain framework clone the repository found on: *https://github.com/erix5son*. If you don't find the repository on the website, write to the next email: *erikgrabljevec5@gmail.com*. In repository you will find files *setup.py*, *settings.py*, few other files and multiple folders. In *settings.py* you must set the root directory of the project and the database connection. Next install all the dependenies found in *requirements.txt*. Finally, run the command *sudo setup.py install* to install the modules. To get a grasp on how the framework is used, check the examples in the *demos* folder.

Database

As mentioned, database connection should be set in *settings.py*. Once this has been set, one can build the database by simply running *build_database.py* found in folder *data_tools*. In folder *data_tools* there is one more script *handle_data.py* containing functions to obtain the data from the database in Pandas dataframe. Furthermore, there are functions to modify data obtained.

Ranking systems

In folder *ranking_systems* there are multiple ranking systems implemented, together with tests to confirm that the equations used are correct. They have a straightfor-

ward structure and reading through the code is enough to start using them.

Calculations

Inside folder *models/calculations* there is a script *calculations.pyx*. Here are functions that are often called and must be implemented efficiently. To optimize calculations libraries *Cython* and *Numpy* are used. For user two functions are very important. *prob_win_match(p, q, best_of=3)* returns probability of player winning, where first player one and player two have probabilities of winning point on serve p and q in a best of three or five match. Second function is *monte_carlo_match(p, q, best_of=3, N=100000)* which uses monte carlo to calculate expected result and expected number of points played in match.

Model

We implement model by deriving from class $TennisRankingModel$. Implement method *run* and *train_params*. There is very important thing to note and that is that you must use *self.players_ranking* for player rankings and safe all updates in it at the end of *run* method. The easiest way to use it is to mimic already implemented classes.

Once derived model is implemented use function *analyse_ranking_model* on it. It will write out report on model performance in console. Latex version of analysis will be stored in *reports* folder.

Graphical tool

Implemented using *flask*. To use it run the command *python server.py* from the *graphical_tool* folder.