

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	In-game Sports Analytics . . . . .	3
1.2	History of Tennis Forecasting . . . . .	4
1.3	Match/Point-by-Point Datasets . . . . .	4
1.4	Implementation . . . . .	5
<b>2</b>	<b>Scoring</b>	<b>6</b>
2.1	Modeling games . . . . .	6
2.2	Modeling sets . . . . .	7
2.3	Modeling a tie-break game . . . . .	8
2.4	Modeling a best-of-three match . . . . .	8
2.5	Win Probability Equation . . . . .	9
<b>3</b>	<b>Pre-game Match Prediction</b>	<b>10</b>
3.1	Overview . . . . .	10
3.2	Elo Ratings . . . . .	10
3.3	ATP Rank . . . . .	11
3.4	Point-based Model . . . . .	11
3.5	James-Stein Estimator . . . . .	13
3.6	Opponent-Adjusted Serve/Return Statistics . . . . .	15
3.7	Results . . . . .	15
3.7.1	Table . . . . .	15
3.7.2	Discussion . . . . .	16
<b>4</b>	<b>In-game Match Prediction</b>	<b>18</b>
4.1	Logistic Regression . . . . .	18
4.1.1	Cross Validation . . . . .	19
4.2	Random Forests . . . . .	20
4.3	Hierarchical Markov Model . . . . .	20
4.3.1	Beta Experiments . . . . .	20
4.3.2	Elo-induced Serve Probabilities . . . . .	21
4.3.3	Significance of $f_{ij} + f_{ji}$ . . . . .	22
4.4	Results . . . . .	23
4.4.1	Model Calibration . . . . .	25
4.4.2	By Set . . . . .	26
4.4.3	By Surface . . . . .	27

4.4.4	Distribution of Performance . . . . .	27
4.4.5	Variable Importance . . . . .	27
4.5	Visualizing Win Probability . . . . .	28
<b>5</b>	<b>Conclusion</b>	<b>30</b>
5.1	Real-World Applications . . . . .	30
5.2	Future Steps . . . . .	31
5.3	Visualizing Grand Slam Matches . . . . .	31
5.3.1	Roger Federer vs. Rafael Nadal Australian Open 2017 Final . . . . .	31
5.3.2	Novak Djokovic vs. Roger Federer US Open 2011 Semi-Final . . . . .	32

# 1. Introduction

## 1.1 In-game Sports Analytics

“The win probability graphic/discussion on ESPN is literally taking a sword and sticking it through the chest of any fun left in baseball”

— Kenny Ducey (@KennyDucey) April 2, 2017

“ESPN showing win probability is extremely good. Next up, run expectancy!”

— Neil Weinberg (@NeilWeinberg44) April 4, 2017

“Thank the Lord we have the ESPN Win Probability stat to tell us the team that’s ahead has a good chance to win.”

— Christian Schneider (@Schneider\_CM) April 17, 2017

In recent years, win probability has become increasingly prevalent in sports broadcasting. Brian Burke, an NFL commentator, has posted live win-probability graphs of NFL playoff games on his website for the past few years <sup>1</sup>. Earlier this year, ESPN began to post win probabilities atop the score box in televised Major League Baseball games [?]. Despite mixed reactions from fans, as shown above, these developments represent a transition in sports broadcasting’s modern narrative.

A win probability from any score line communicates how much a team or player is favored to win. While one can produce this from any model of choice, those in sports analytics strive to produce the most well-informed estimate available. With the recent proliferation of online betting, in-match win probability dictates an entire market of its own. Betfair’s platform matched over 40 million Euros during the 2011 French Open Final [6]; other high-profile matches often draw comparable volume over betting exchanges. While the majority of tennis prediction papers concern pre-match prediction, around 80% of online betting occurs while matches are in progress [10]. At the very least, in-match win probability concerns all participants in this betting market.

---

<sup>1</sup><http://www.advancedfootballanalytics.com/index.php/home/tools/live-playoff-probabilities>

## 1.2 History of Tennis Forecasting

Statistical prediction models have been applied to tennis matches for over twenty years. Klaassen and Magnus tested the assumption that points in a tennis match are independent and identically distributed [7]. While the assumption is false, they concluded that deviations are small enough that constant probabilities provide a reasonable approximation. Under this assumption, they construct a hierarchical Markov model in conjunction with tennis' scoring system. From this model, they offer an analytical equation for match-win probability from any score, given each individual player's probability of winning a point on serve [8]. Barnett and Clarke then offer a method of estimating each player's serve probability from historical data [1]. Years later, Bevc proposed updating each of these serve probabilities with a beta distribution between each point and computing the corresponding win probability with the above model [4]. Recently, Kovalchik assessed performance of 11 different pre-match prediction models on 2014 ATP tour-level matches. While the market-based Bookmaker Consensus Model performed best overall, Elo ratings proved most effective among methods that rely on historical data [9].

Over the past several years, Jeff Sackmann has released the largest publicly available library of tennis datasets via github. This collection contains match summaries of every ATP and WTA match in the Open Era, point-by-point summaries of nearly 100,000 tour-level and satellite matches, and a crowd-sourced match-charting project spanning over 3000 matches, where volunteers record each shots type and direction over an entire match. While FiveThirtyEight and Kovalchik have used Jeff Sackmanns match data to generate elo ratings, none of the aforementioned papers have tested models with his point-by-point dataset.

Paragraph setting up 1vs1 in tennis...

As past papers have spanned several decades, datasets and model evaluation are not consistent. Klaassen and Magnus' original point-based model and Bevc's beta experiments both apply methods to around 500 Wimbledon matches from 1991-94. Barnett explores point-based models in great depth, yet primarily applies them to a single marathon match between Andy Roddick and Younes El Aynaoui from the 2002 Australian Open [3]. Except for Bevc, none of these papers assess in-game match prediction with metrics such as accuracy or cross-entropy. While Bevc does record accuracy of results over 500 Wimbledon matches, he only reports accuracy of predictions in the third set and onwards, a subset of the entire dataset. In other words, no one has taken all in-game prediction models and tested them at a large scale. For this reason, I will test all relevant in-match prediction methods on thousands of matches within the past seven years.

This paper combines Elo ratings, a wealth of data, and current technology to provide a similar survey of which in-match prediction methods perform the best. I build upon past research by testing variations of previous state-of-the-art methods, and applying new concepts to these datasets, from probability models used in football to elo-induced serving percentages.

## 1.3 Match/Point-by-Point Datasets

This project uses two different types of datasets: one with match summary statistics and one with point-by-point information. Both are publicly available on github, courtesy of Jeff Sackmann (<https://github.com/JeffSackmann>). We primarily use the matches dataset under

“`tennis_atp`” to test pre-match prediction methods. This dataset covers over 150,000 tour-level matches, dating back to 1968. Features include player qualities, such as nationality or dominant hand, as well as match statistics, like serve/return points won.

The data in “`tennis_pointbypoint`” offers more granular detail about a single match’s point progression. Each match contains a string listing the order in which players won points and switched serve. As the `atpworldtour.com` does not publicly list point-by-point summaries, this information was presumably extracted from a betting website.

## 1.4 Implementation

With the above datasets serving as a basis for this project, thorough re-formatting of the data was required in order to connect point-by-point strings to their corresponding rows in the match dataset. As both datasets included player names, year, and score, we connected matches across datasets with a hashing scheme <sup>2</sup>. Due to observed inconsistencies, we canonicalized player names between match and point-by-point datasets <sup>3</sup> in order to maximize the number of available matches with point-by-point summaries. While a portion of the point-by-point summaries are from satellite events, we were able to match around 10,600/12,000 point-by-point strings to their respective tour-level matches. Then, we could associate information generated from the match dataset’s entirety (elo ratings, year-long adjusted serve stats, etc) with corresponding point-by-point strings.

To view this process in more depth or access any of the resulting datasets, visit [https://github.com/jgollub1/tennis\\_match\\_prediction](https://github.com/jgollub1/tennis_match_prediction). Implementations of each method in this project, and instructions on how to generate all relevant statistics, are also provided.

the csv’s are gigantic, so should I provide an abbreviated diagram???

---

<sup>2</sup>eg. `hash(matchX)` = “Roger Federer Tomas Berdych 2012 3-6 7-5 7-5”

<sup>3</sup>eg. “Stan Wawrinka” → “Stanislas Wawrinka”, “Federico Del Bonis” → “Federico Delbonis”

## 2. Scoring

Tennis' scoring system consists of three levels: sets, games, and points. Consider a tennis match between two entities,  $p_i$  and  $p_j$ . We can represent any score as  $(s_i, s_j, g_i, g_j, x_i, x_j)$  where  $i$  is serving and  $s_k, g_k, x_k$  represent  $p_k$ 's score in sets, games, and points, respectively <sup>1</sup>. The players alternate serve each game and continue until someone clinches the match by winning two sets (best-of-three) or three sets (best-of-five) <sup>2</sup>.

The majority of in-play tennis models utilize a graphical structure that embodies the levels within tennis' scoring system. Madurska referred to this as a hierarchical Markov Model [?]. Barnett formally defines this representation for tennis scores [2]. With  $p_i$  and  $p_j$  winning points on serve with probabilities  $f_{ij}, f_{ji}$ , each in-match scoreline  $(s_i, s_j, g_i, g_j, x_i, x_j)$  progresses to one of its two neighbors  $(s_i, s_j, g_i, g_j, x_i + 1, x_j)$  and  $(s_i, s_j, g_i, g_j, x_i, x_j + 1)$  with transition probabilities dependent on the current server. Assuming all points in a match are i.i.d. between servers, we can then use the below model to recursively determine win probability. Consider a match with  $p_i$  serving at 7-5 5-3 30-30. Representing the score in terms of  $s_k, g_k, x_k$ , we find:

$$P_m(1, 0, 5, 3, 2, 2) = P(p_i \text{ wins the match when serving from this scoreline})$$

$$P_m(1, 0, 5, 3, 2, 2) = f_{ij} * P_m(1, 0, 5, 3, 3, 2) + (1 - f_{ij})P_m(1, 0, 5, 3, 2, 3)$$

In the following sections, we specify boundary values to each level of our hierarchical model.

### 2.1 Modeling games

Within a game, either  $p_i$  or  $p_j$  serves every point. Every game starts at (0,0) and to win a game, a player must win four or more points by a margin of at least two. Consequently, all games with valid scores  $(x_i, x_j)$  where  $x_i + x_j > 6$ ,  $|x_i - x_j| \leq 1$  are reduced to (3,3), (3,2), or (2,3). Furthermore, the win probability at (3,3) can be calculated directly. From (3,3), the server wins the next two points with probability  $f_{ij}^2$ , the returner wins the next two points with probability  $(1 - f_{ij})^2$ , or both players split the two points and return to (3,3) with probability  $2f_{ij}(1 - f_{ij})$ . Relating the game's remainder to a geometric sequence, we find  $P_g(3, 3) = \frac{f_{ij}^2}{f_{ij}^2 + (1 - f_{ij})^2}$ . As the probability of winning each point remains constant for both players, this graph assumes that points within a game are independent.

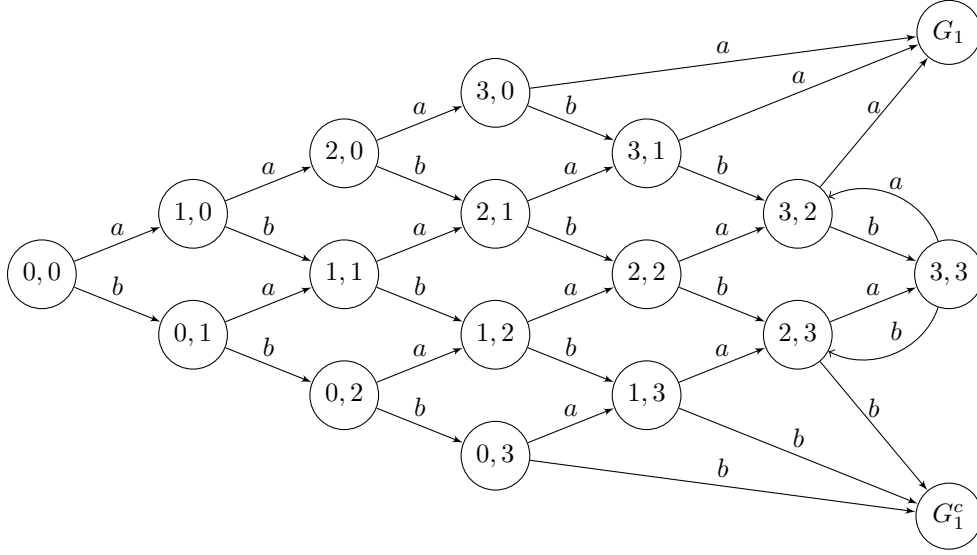
<sup>1</sup>While tennis officially refers to a game's first three points as 15,30,40 we will call them 1,2,3 for simplicity's sake

<sup>2</sup>The best-of-five format is typically reserved for men's grand slam and Davis Cup events

Possible sequences of point scores in a game:

a - player  $i$  wins the following point

b - player  $j$  wins the following point



Boundary values:

$$P_g(x_i, x_j) = \begin{cases} 1, & \text{if } x_1 = 4, x_2 \leq 2 \\ 0, & \text{if } x_2 = 4, x_1 \leq 2 \\ \frac{f_{ij}^2}{f_{ij}^2 + (1 - f_{ij})^2}, & \text{if } x_1 = x_2 = 3 \\ f_{ij} * P_g(s_i, s_j, g_i, g_j, x_i + 1, x_j) + (1 - f_{ij})P_g(s_i, s_j, g_i, g_j, x_i, x_j + 1), & \text{otherwise} \end{cases} \quad (2.1)$$

With the above specifications, we can efficiently compute player  $i$ 's win probability from any score  $P_g(x_i, x_j)$ .

## 2.2 Modeling sets

Within a set,  $p_i$  or  $p_j$  alternate serve every game. Every set starts at (0,0). To win a set, a player must win six or more games by a margin of at least two. If the set score (6,6) is reached, a special tiebreaker game is played to determine the outcome of the match.

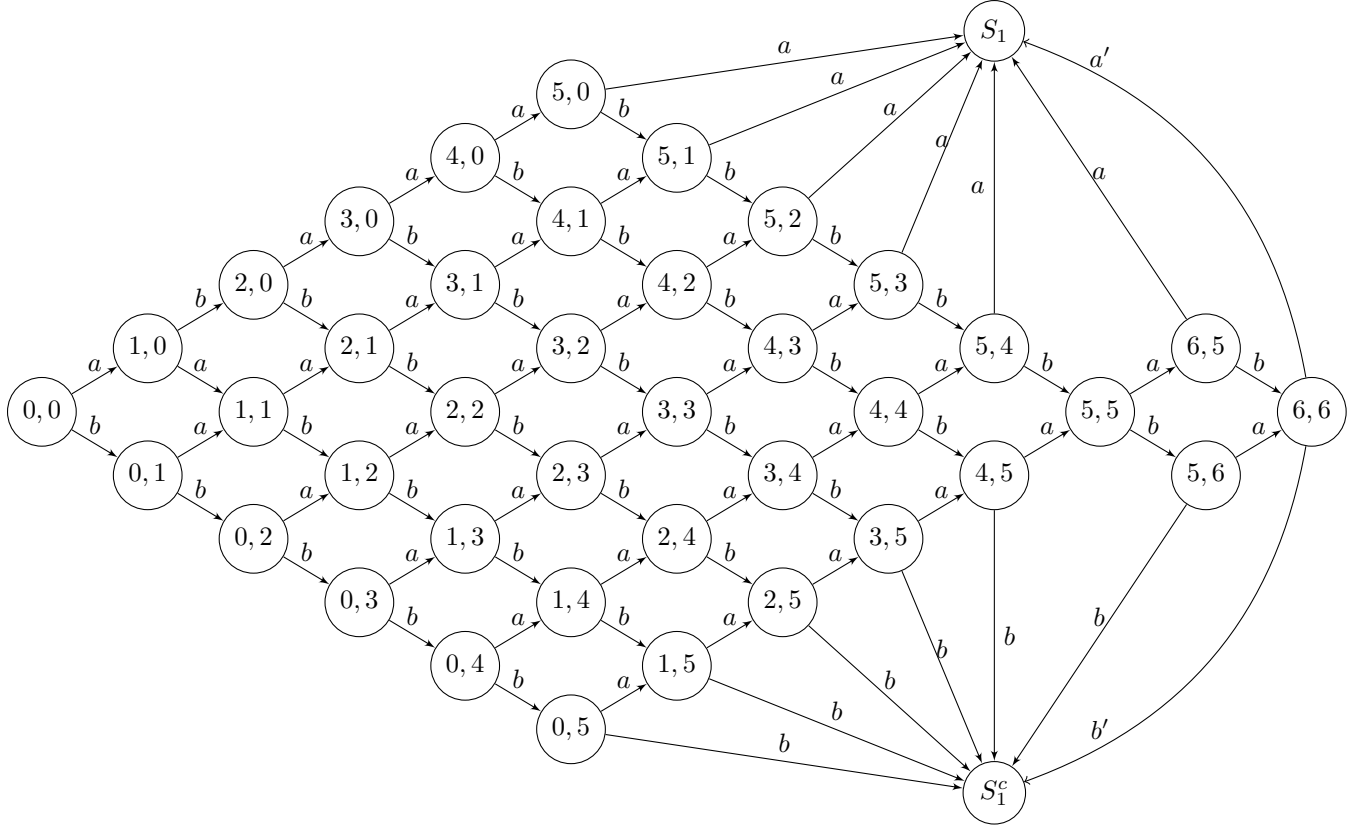
Possible sequences of point scores in a game:

a - player 1 wins the following game

b - player 2 wins the following game

$a'$  - player 1 wins the tiebreaker game

$b'$  - player 2 wins the tiebreaker game



Boundary values:

$$P_s(g_1, g_2) = \begin{cases} 1, & \text{if } g_1 \geq 6, g_1 - g_2 \geq 2 \\ 0, & \text{if } g_2 \geq 6, g_2 - g_1 \geq 2 \\ P_{tb}(s_1, s_2), & \text{if } g_1 = g_2 = 6 \\ P_g(0, 0)(1 - P_s(g_2, g_1 + 1)) + (1 - P_g(0, 0))(1 - P_s(g_2 + 1, g_1)), & \text{otherwise} \end{cases} \quad (2.2)$$

See appendix for the tiebreak game's corresponding diagram\*\*

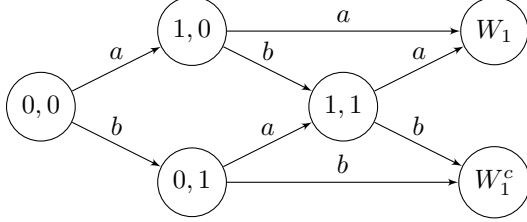
## 2.3 Modeling a tie-break game

## 2.4 Modeling a best-of-three match

a - player 1 wins the following set



b - player 2 wins the following set



Boundary values:

$$P_m(s_1, s_2) \begin{cases} 1, & \text{if } g_1 \geq 2 \\ 0, & \text{if } g_2 \geq 2 \\ P_s(0,0)(P_m(s_1 + 1, s_2)) + (1 - P_s(0,0))(P_m(s_1, s_2 + 1)), & \text{otherwise} \end{cases} \quad (2.3)$$

## 2.5 Win Probability Equation

Combining the above equations, we can recursively calculate  $p_i$ 's win probability when serving from  $(s_i, s_j, g_i, g_j, x_i, x_j)$  as:

$$P_m(s_i, s_j, g_i, g_j, x_i, x_j) = f_{ij} * P_m(s_i, s_j, g_i, g_j, x_i + 1, x_j) + (1 - f_{ij})P_m(s_i, s_j, g_i, g_j, x_i, x_j + 1)$$

Unless  $(x_i, x_j)$  satisfies a boundary condition,  $P_m(s_i, s_j, g_i, g_j, x_i, x_j)$  will recursively call itself on the two possible following scores. Since we can compute  $P_g(3, 3)$ , we can eventually represent all each recursive probabilities in terms of boundary values, which allows us to compute the probability of winning the current game. By traversing through the set and match models in similar recursive fashion, we can eventually determine  $p_i$ 's probability of winning the match.

## 3. Pre-game Match Prediction

### 3.1 Overview

Before play has started, an in-match prediction model cannot draw on information from the match itself. Then, before a match between  $p_i$  and  $p_j$  commences, the most well-informed pre-match forecast  $\hat{\pi}_{ij}(t)$  should serve as a basis for prediction. Therefore, we first explore pre-match models as a starting point for in-match prediction.

Earlier this year, Kovalchik released a survey of eleven different pre-match prediction models, assessing them side-by-side in accuracy, log-loss, calibration, and discrimination. 538's Elo-based model and the Bookmaker Consensus Model performed the best. Elo-based prediction incorporates  $p_i$  and  $p_j$ 's entire match histories, while the BCM model incorporates information encoded in various betting markets. In this section, we explore pre-match prediction methods that do not employ market data.

Road map:

### 3.2 Elo Ratings

Elo was originally developed as a head-to-head rating system for chess players (1978). Recently, 538's Elo variant has gained prominence in the media. For a match at time  $t$  between  $p_i$  and  $p_j$  with Elo ratings  $E_i(t)$  and  $E_j(t)$ ,  $p_i$  is forecasted to win with probability:

$$\hat{\pi}_{ij}(t) = (1 + 10^{\frac{E_j(t) - E_i(t)}{400}})^{-1}$$

$p_i$ 's rating for the following match  $t + 1$  is then updated accordingly:

$$E_i(t + 1) = E_i(t) + K_{it} * (W_i(t) - \hat{\pi}_{ij}(t))$$

$W_i(t)$  is an indicator for whether  $p_i$  won the given match, while  $K_{it}$  is the learning rate for  $p_i$  at time  $t$ . According to FiveThirtyEight's analysts, Elo ratings perform optimally when allowing  $K_{it}$  to decay slowly over time [?]. With  $m_i(t)$  representing the  $p_i$ 's career matches played at time  $t$  we update our learning rate:

$$K_{it} = \frac{250}{(5 + m_i(t))^4}$$

This variant updates a player's elo most quickly when we have no information about a player and makes smaller changes as  $m_i(t)$  accumulates. To apply this elo rating method to our

dataset, we initialize each player’s elo rating at  $E_i(0) = 1500$  and match history  $m_i(0) = 0$ . Then, we iterate through all tour-level matches from 1968-2017 <sup>1</sup> in chronological order, storing  $E_i(t), E_j(t)$  for each match and updating each player’s elo accordingly.

### 3.3 ATP Rank

While Klaassen incorporated ATP rank into his prediction model [8], Kovalchik and 538 concur that elo outperforms ranking-based methods. On ATP match data from 2010-present, we found:

method	accuracy
ATP Rank	66.5 %
Standard elo	69.0 %

With each rating system, we forecast the higher-ranked player as the winner. Accuracy represents the proportion of correctly forecasted matches. As the ATP does not provide fans a corresponding win-probability equation to go along with their ranking system, we cannot compare the approaches in terms of log loss or calibration. Still, considering its superior accuracy to ATP rank in recent years, models in this paper use elo ratings to represent a player’s ability in place of official tour rank.

### 3.4 Point-based Model

The hierarchical Markov Model offers an analytical solution to win probability  $\hat{\pi}_{ij}(t)$  between players  $p_i$  and  $p_j$ , given serving probabilities  $f_{ij}, f_{ji}$ . This hinges on the Markov assumption that transition probabilities from any state only depend on the current state. In other words, all points are independent and the probability of winning a given point only depends on the current server. While this is counter-intuitive, Klaassen and Magnus demonstrated that deviations from the iid assumption within matches are small enough to reasonably justify point-based models [7]. Later on, Barnett and Clarke demonstrated a method to estimate  $f_{ij}, f_{ji}$ , given players’ historical serve/return averages [1].

$$\begin{aligned} f_{ij} &= f_t + (f_i - f_{av}) - (g_j - g_{av}) \\ f_{ji} &= f_t + (f_j - f_{av}) - (g_i - g_{av}) \end{aligned}$$

Each player’s serve percentage is a function of their own serving ability and their opponent’s returning ability.  $f_t$  denotes the average serve percentage for the match’s given tournament, while  $f_i, f_j$  and  $g_i, g_j$  represent  $p_i$  and  $p_j$ ’s percentage of points won on serve and return, respectively.  $f_{av}, g_{av}$  are tour-level averages for serve and return percentage. Since all points are won by either server or returner,  $f_{av} = 1 - g_{av}$ . By combining differences in ability relative to the average This formula assumes that their effects are additive.

As per Barnett and Clarke’s formula, we use the previous year’s tournament serving statistics to calculate  $f_t$  for a given tournament and year, where  $(w, y)$  represents the set of all matches

---

<sup>1</sup>tennis’ Open Era began in 1968, when professionals were allowed to enter grand slam tournaments. Before then, only amateurs played these events

played at tournament  $w$  in year  $y$ .

$$f_t(w, y) = \frac{\sum_{k \in (w, y-1)} \# \text{ of points won on serve in match } k}{\sum_{k \in (w, y-1)} \# \text{ of points played in match } k}$$

In their paper, Barnett and Clarke only apply this method to a single match: Roddick vs. El Aynaoui Australian Open 2003. Furthermore, their ability to calculate serve and return percentages is limited by aggregate statistics supplied by atpworldtour.com. That is, they can only use year-to-date serve and return statistics to calculate  $f_i, g_i, f_j, g_j$ . Since the statistics do not list corresponding sample sizes, they must assume that each best-of-three match lasts 165 points, which adds another layer of uncertainty in estimating players' abilities.

Implementing this method with year-to-date statistics proves troublesome because  $f_i, g_i$  decrease in uncertainty as  $p_i$  accumulates matches throughout the year. Due to availability of data, match forecasts in September will then be far more reliable than ones made in January. However, with our tour-level match dataset, we can keep a year-long tally of serve/return statistics for each player at any point in time. Where  $(p_i, y, m)$  represents the set of  $p_i$ 's matches in year  $y$ , month  $m$ , we obtain the following statistics <sup>2</sup>:

$$f_i(y, m) = \frac{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} \# \text{ of points won on serve by } i \text{ in match } k}{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} \# \text{ of points played on serve by } i \text{ in match } k}$$

$$g_i(y, m) = \frac{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} \# \text{ of points won on return by } i \text{ in match } k}{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} \# \text{ of points played on return by } i \text{ in match } k}$$

Keeping consistent with this format, we also calculate  $f_{av}, g_{av}$  where  $(y, m)$  represents the set of tour-level matches played in year  $y$ , month  $m$ :

$$f_{av}(y, m) = \frac{\sum_{t=1}^{12} \sum_{k \in (y-1, m+t)} \# \text{ of points won on serve in match } k}{\sum_{t=1}^{12} \sum_{k \in (y-1, m+t)} \# \text{ of points played in match } k} = 1 - g_{av}(y, m)$$

Now, variance of  $f_i, g_i$  no longer depends on time of year. Since the number of points won on serve are recorded in each match, we also know the player's number of serve/return points played. Below, we combine player statistics over the past 12 months to produce  $f_{ij}, f_{ji}$  for Kevin Anderson and Fernando Verdasco's 3rd round match at the 2013 Australian Open.

player name	s_points_won	s_points	$f_i$	r_points_won	r_points	$g_i$
Kevin Anderson	3292	4842	.6799	1726	4962	.3478
Fernando Verdasco	2572	3981	.6461	1560	4111	.3795

From 2012 Australian Open statistics,  $f_t = .6153$ . From tour-level data spanning 2010-2017,  $f_{av} = 0.6468$ ;  $g_{av} = 1 - f_{av} = .3532$  Using the above serve/return statistics from 02/12-01/13, we can calculate:

$$f_{ij} = f_t + (f_i - f_{av}) - (g_j - g_{av}) = .6153 + (.6799 - .6468) - (.3795 - .3532) = .6221$$

$$f_{ji} = f_t + (f_j - f_{av}) - (g_i - g_{av}) = .6153 + (.6461 - .6468) - (.3478 - .3532) = .6199$$

With the above serving percentages, Kevin Anderson is favored to win the best-of-five match with probability  $M_p(0, 0, 0, 0, 0) = .5139$

---

<sup>2</sup>for the current month  $m$ , we only collect month-to-date matches

### 3.5 James-Stein Estimator

Decades ago, Efron and Morris described a method to estimate groups of sample means [5]. The James-Stein estimator shrinks sample means toward the overall mean, with shrinkage proportional to its estimator's variance. Regardless of the value of  $\theta$ , this method produces results superior in expectation to the MLE method, an admissible estimator.

To estimate serve/return parameters for players who do not regularly play tour-level events,  $f_i, g_i$  must be calculated from limited sample sizes. Consequently, match probabilities based off these estimates may be skewed by noise. The James-Stein estimators offer a more reasonable estimate of serve and return ability for players with limited match history.

To shrink serving percentages, we compute the variance of all recorded  $f_i$  statistics <sup>3</sup> in our match data set  $D_m$ .

$$\hat{\tau}^2 = \sum_{f_i \in D_m} (f_i - f_{av})^2$$

Then, each estimator  $f_i$  is based off  $n_i$  service points. With each estimator  $f_i$  representing  $f_i/n_i$  points won on serve, we can compute estimator  $f_i$ 's variance and a corresponding normalization coefficient:

$$\hat{\sigma}_i^2 = \frac{f_i(1-f_i)}{n_i}$$

$$B_i = \frac{\hat{\sigma}_i^2}{\hat{\tau}^2 + \hat{\sigma}_i^2}$$

Finally, the James-Stein estimator takes the form:

$$JS(f_i) = f_i + B_i(f_{av} - f_i)$$

We repeat the same process with  $g_i$  to obtain James-Stein estimators for return statistics.

To see how shrinkage makes our model robust to small sample sizes, consider the following example. When Daniel Elahi (COL) and Ivo Karlovic (CRO) faced off at ATP Bogota 2015, Elahi had played only one tour-level match in the past year. From a previous one-sided victory, his year-long serve percentage,  $f_i = 51/64 = .7969$ , was abnormally high compared to the year-long tour-level average of  $f_{av} = .6423$ .

player name	s_points_won	s_points	$f_i$	r_points_won	r_points	$g_i$	elo rating
Daniel Elahi	51	64	.7969	22	67	.3284	1516.9178
Ivo Karlovic	3516	4654	.7555	1409	4903	.2874	1876.9545

$$f_{ij} = f_t + (f_i - f_{av}) - (g_j - g_{av}) = .6676 + (.7969 - .6423) - (.2874 - .3577) = .8925$$

$$f_{ji} = f_t + (f_j - f_{av}) - (g_i - g_{av}) = .6676 + (.7555 - .6423) - (.3284 - .3577) = .8101$$

Following Klaassen and Magnus' method of combining player outcomes, we compute Elahi's service percentage to be 89.3%. This is extremely high, and eclipses Karlovic's 81.01% serve projection. This is strange, given that Karlovic is one of the most effective servers in the history of the game. From the serving stats, our hierarchical Markov Model computes Elahi's win probability as  $M_p(0,0,0,0,0,0) = .8095$ . This forecast seems unreasonably confident of

---

<sup>3</sup>each  $f_i$  is computed from the previous twelve months of player data

Elahi's victory, despite only having collected his player statistics for one match. Karlovic's 360-point elo advantage calculates Elahi's win probability as

$$\hat{\pi}_{ij}(t) = (1 + 10^{\frac{1876.9545 - 1516.9178}{400}})^{-1} = .1459$$

which leads us to further question the validity of this approach when using limited historical data. Thus, we turn to the James-Stein estimator to normalize Elahi's serving and return probabilities.

$$JS(f_i) = f_i + B_i(f_{av} - f_i) = .7969 + .7117(.6423 - .7969) = .6869$$

$$JS(g_i) = g_i + B_i(g_{av} - g_i) = .3284 + .7624(.3577 - .3284) = .3507$$

$$JS(f_j) = f_j + B_j(f_{av} - f_j) = .7555 + .0328(.6423 - .7555) = .7518$$

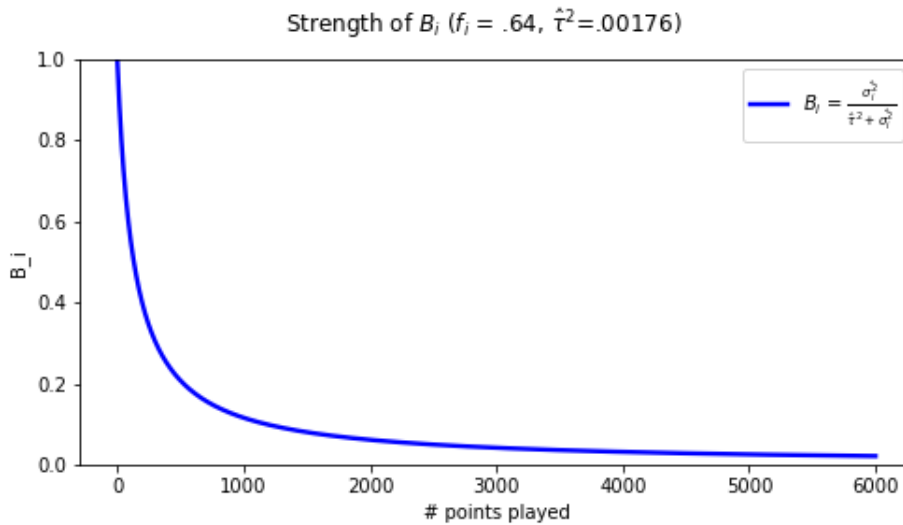
$$JS(g_j) = g_j + B_j(g_{av} - g_j) = .2874 + .0420(.3577 - .2874) = .2904$$

$$JS(f_{ij}) = f_t + (JS(f_i) - f_{av}) - (JS(g_j) - g_{av}) = .6676 + (.6869 - .6423) - (.2904 - .3577) = .7795$$

$$JS(f_{ji}) = f_t + (JS(f_j) - f_{av}) - (JS(g_i) - g_{av}) = .6676 + (.7518 - .6423) - (.3507 - .3577) = .7841$$

Above, we can see that the James-Stein estimator shrinks Elahi's stats far more than Karlovic's, since Karlovic has played many tour-level matches in the past year. Given  $JS(f_i)$ ,  $JS(f_j)$ , we compute  $M_p(0, 0, 0, 0, 0) = .4806$ . By shrinking the serve/return statistics, our model lowers Elahi's inflated serve percentage and becomes more robust to small sample sizes.

As point-based forecasts on limited data threaten model performance, especially with respect to cross entropy, the James-Stein estimator allows a safer way to predict match outcomes. Later on, we will use the James-Stein estimator to normalize not only year-long serve/return statistics, but also surface-specific and opponent-adjusted percentages. For a player who has averaged  $f_i = .64$  over  $n$  points in the past twelve months, this diagram illustrates how normalization coefficient  $B_i$  varies with  $n$ :



### 3.6 Opponent-Adjusted Serve/Return Statistics

While Barnette and Clarke’s equation does consider opponent’s serve and return ability, it does not track average opponents’ ability within a player’s history. This is important, as a player’s serve/return percentages may become inflated from playing weaker opponents or vice versa. In this section, we propose a variation to Barnette and Clarke’s equation which replaces  $f_{av}, g_{av}$  with opponent-adjusted averages  $1 - g_{i\_opp\_av}, 1 - f_{i\_opp\_av}$  for  $p_i$ . The equations then become:

$$\begin{aligned} f_{ij} &= f_t + (f_i - (1 - g_{i\_opp\_av})) - (g_j - (1 - f_{j\_opp\_av})) \\ f_{ji} &= f_t + (f_j - (1 - g_{j\_opp\_av})) - (g_i - (1 - f_{i\_opp\_av})) \end{aligned}$$

$g_{i\_opp\_av}$  represents the average return ability of opponents that  $p_i$  has faced in the last twelve months. To calculate this, we weight each opponent’s return ability  $g_j$  by number of points in their respective match.

$$g_{i\_opp\_av} = \frac{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} (\# \text{ of } p_i \text{'s return points in match } k) * (p_j \text{'s return ability in match } k)}{\sum_{t=1}^{12} \sum_{k \in (i, y-1, m+t)} \# \text{ of } p_i \text{'s return points in match } k}$$

### 3.7 Results

The following results were obtained from testing methods on 2014 ATP best-of-three matches, excluding Davis Cup. There were 2409 matches in this dataset.

#### 3.7.1 Table

We observe performance across variants of elo-based predictions and point-based models. Since all original implementations provide explicit formulas with no optimization, we directly assess their performance on 2014 tour-level match data. In the case of logistic regression, the model was trained on tour-level match data from 2011-2013. We measure each model’s performance in terms of accuracy, as in section 3.3. We also consider log loss, a measure of overall likelihood of a model’s predictions. Log loss is important because it penalizes models that incorrectly assign over-confident or under-confident win probabilities. As we are hoping to optimize in-match prediction models’ log loss, it is important to consider the likelihood of various pre-match forecasts. The following terms express variations to point-based and elo models:

**KM** - point-based hierarchical Markov models with combined serve/return percentages from Barnett/Clarke

**James-Stein** - version of a KM model, with all serve/return percentages normalized with James-Stein estimator

**surface** - version of model where all ratings and percentages are stratified by surface (Hard, Clay, Grass)

**538** - specifically denotes Five-Thirty-Eight’s “decaying K-factor” method in computing elo

method	accuracy	log loss
KM	64.8	.649
KM James-Stein	65.4	.616
KM surface	63.3	.707
KM surface James-Stein	63.6	.639
KM adjusted	67.8	.632
KM adjusted James-Stein	67.9	.617
elo	69.1	.586
surface elo	68.4	.591
elo 538	69.2	.587
surface elo 538	69.4	.592
logit (elo 538, surface elo 538)	69.5	.577

### 3.7.2 Discussion

As expected, James-Stein normalization significantly improves each point-based model’s log loss. While surface-based elo is competitive with regular elo, restricting Klaassen and Magnus’ point-based method to surface-specific data clearly hurts performance. This is presumably due to limited surface-specific data. As many players perform differently across surfaces, computing serve percentages from surface-specific data offers the potential to express how players’ performance varies by surface. Yet, even with James-Stein normalization, the loss in sample size appears to outweigh the benefits obtained from surface-specific data. As we did not rigorously explore possibilities for weighting players’ surface-specific statistics, there may still be potential for a point-based model that effectively considers surface in computing serve/return percentages. Finally, using opponent-adjusted serve-return statistics further improved performance. By fully incorporating each opponent’s ability in a player’s match history, this model approached elo ratings in likeness of predictions, with win-probability forecasts between the two sharing  $R^2 = .76$ .

By plugging elo and surface elo into a logistic regression model, we achieve a log loss of .577. After fitting to the training data, this model learned coefficients for each variable that assigned a 56% weight to regular elo ratings and 44% weight to surface elo ratings. The near-even weight to both types of ratings suggests how important surface-specific ability is toward forecasting matches. Aside from models which draw information directly from the betting market, no other model has documented preferable log loss. Kovalchik reported 70% accuracy using 538’s elo method [9], calculating elo ratings using satellite events (ATP Challengers and Futures) in addition to tour-level events <sup>4</sup>. While this accounted for a small increase in accuracy, her method achieved a log loss of .59, which does not outperform our implementation. For simplicity’s sake, we calculated all elo ratings and statistics from tour-level matches alone <sup>5</sup>. In the end, our elo-based logistic regression’s incremental improvement over singular elo ratings implies that an effective rating system should consider a overall ability as well as surface-specific ability.

Sipko recently explored machine-learning methods for pre-match prediction, surveying logistic regression, the common-opponent model, and an artificial neural net [10]. While he claimed to have achieved 4.3% return-on-investment off the betting market with an artificial neural

<sup>4</sup>one can observe at <https://github.com/skoval/deuce>

<sup>5</sup>there are about 3000 tour-level matches each year



net, his machine learning models did not beat a log loss of .61 when predicting 2013-2014 ATP matches <sup>6</sup>. Despite the current deep-learning craze throughout industry and academia (cite nyt article), no one has published findings to suggest their superiority in predicting tennis matches. Short of market-based models, it appears most effective to stick with elo’s “ruthlessly Bayesian design” which Five-Thirty-Eight so frequently touts (cite: article). Transitioning onward, we will consider how these findings may inform an effective in-match prediction model.

---

<sup>6</sup>the models were trained on 2004-2010 ATP matches and validated on 2011-2012 ATP matches

## 4. In-game Match Prediction

The following methods will be tested on tour-level matches for which we have point-by-point data. The matches span 2010-2017, accounting for nearly half of all tour-level matches within this time. Point-by-point records in Sackmann’s dataset take the form of the following string:

Mikhail Youzhny vs. Evgeny Donskoy (Australian Open 2013, Round 2)

$P = \text{“}SSRSS; RRRR; SRSSS; SRRSRSSS; SRSSRS; RSRSSS; SRSRSS; RSRSRSSS; SSS S.SSSRRRSS; RSSSS; SSRSS; SSSRS; SSSS; RRRSSSSRRSSRRSSSS; SRSRSS; SSS RS; RSRSSRSS; SSSS; SRSSS; RSRSSRRSSS; R/SR/SS/RR/RS/SR.RSRRR; \dots\text{”}$

$S$  denotes a point won by the server and  $R$  a point won by the returner. Individual games are separated by “;” sets by “.” and service changes in tiebreaks by “/”. By iterating through the string, one can construct  $n$  data points  $\{P_0, P_1, \dots, P_{n-1}\}$  from a match with  $n$  total points, with  $P_i$  representing the subset of the match after which  $i$  points have been played.

$P_0 = \text{“”}$

$P_1 = \text{“}S\text{”}$

$P_2 = \text{“}SS\text{”}$

$P_3 = \text{“}SSR\text{”}$

...

With  $M = \{M_1, M_2, \dots, M_k\}$  representing complete match-strings in our point-by-point data set, the size of our enumerated data set becomes  $\sum_{i=1}^k |M_i|$ . In total, this sums to over 1.2 million individual points within ATP matches spanning 2010-2017. While many in-match prediction models utilize the hierarchical Markov Model structure, we will first test several machine-learning methods as a baseline.

### 4.1 Logistic Regression

Consider a logistic regression model where each observation  $x_i = [x_{i1}, x_{i2}, \dots, x_{ik}]$  has  $k$  features.  $\text{logit}(x_i)$  yields  $p_i$ ’s win probability from input  $x_i$ ,

$$\text{logit}(x_i) = \frac{1}{1 + e^{-(\beta_0 + \beta_1(x_{i1}) + \beta_1(x_{i2}) + \dots + \beta_1(x_{ik}))}}$$

From any scoreline  $(s_i, s_j, g_i, g_j, x_i, x_j)$ , we can simply feed these values into our model as features of  $x_i$  (change notation?). Logistic Regression’s structure makes it easy to consider additional features for each player, such as elo difference, surface elo difference, or break advantage. Before adding more features to the model, we consider two baselines: a model using score differentials  $(s_i - s_j, g_i - g_j, x_i - x_j)$  and another model trained on elo differences and lead heuristic  $L_{ij}$ . This heuristic estimates one  $p_i$ ’s total lead in sets, games, and points:

$$L_{ij} = (s_i - s_j) + \frac{1}{6}(g_i - g_j) + \frac{1}{24}(x_i - x_j)$$

The coefficients preserve order between sets, games, and points, as one cannot lead by six games without winning a set or four points without winning a game. In this model, we consider the following features in predicting the winner of the match:

Table 4.1: Logistic Regression Features

<i>Variable</i>	<i>Description</i>
lead_margin	lead heuristic $L_{ij}$
eloDiff	$\text{Elo}(p_0) - \text{Elo}(p_1)$
s_eloDiff	$\text{s\_Elo}(p_0) - \text{s\_Elo}(p_1)$
setDiff	$\text{SetsWon}(p_0) - \text{SetsWon}(p_1)$
gameDiff	$\text{inSetGamesWon}(p_0) - \text{inSetGamesWon}(p_1)$
pointDiff	$\text{inGamePointsWon}(p_0) - \text{inGamePointsWon}(p_1)$
breakAdv	$\text{ServeGamesWon}(p_0) - \text{ServeGamesWon}(p_1) + \text{I}(\text{currently serving})$
brkPointAdv	$\text{I}(\text{holding break point}) - \text{I}(\text{facing break point})$
sv_points_pct_0	percentage of points won on $p_0$ ’s serve thus far
sv_points_pct_1	percentage of points won on $p_1$ ’s serve thus far

Next, we test the following combinations of features:

- 1) setDiff + gameDiff + pointDiff
- 2) lead\_margin + eloDiff + s\_eloDiff
- 3) all features

#### 4.1.1 Cross Validation

Each match in our best-of-three dataset has around 160 points on average. In tuning hyper parameters for machine learning models, we implement  $k$ -fold group validation with  $k = 5$ . This randomly splits the training set into five subsets of roughly equal size. Treating each subset as a validation set, we train all models on the remaining data before testing performance on the corresponding validation set. Then we average performance across all  $k$  validation sets and select the hyper parameters whose models performed best. By grouping points from the same match together, this method prevents overlap between points from the same match across train, validation, and test sets. Then, when we assess model performance on a test set, our results also reflect performance on matches a model has never seen before.

## 4.2 Random Forests

Brian Burke’s win-probability models are among the most well-known in sports [?]. They calculate a team’s win probability at any point in the match based on historical data through a combination of binning inputs and smoothing their resulting probabilities. Nettleton and Lock built on this method by using a random forest approach.

A random forest consists of an ensemble of randomly generated classification trees. Each tree forms decision functions for a subset of features with splits that generate maximum discriminatory ability. As Nettleton and Lock do so with football-specific features such as down, points, and yards to goal, we do so with analogous tennis features and train on our validation set.

Table 4.2: Random Forest features

Variable	Description
surface	hard, clay, grass
set	first, second, third
eloDiff	$Elo(p_0) - Elo(p_1)$
setDiff	$SetsWon(p_0) - SetsWon(p_1)$
gameDiff	$inSetGamesWon(p_0) - inSetGamesWon(p_1)$
pointDiff	$inGamePointsWon(p_0) - inGamePointsWon(p_1)$
breakAdv	$ServeGamesWon(p_0) - ServeGamesWon(p_1) + I(\text{currently serving})$
brkPointAdv	$I(\text{holding break point}) - I(\text{facing break point})$

## 4.3 Hierarchical Markov Model

With serving percentages already calculated from historical data, our hierarchical Markov model is well-equipped to produce in-match win probability estimates. Using the analytical equation with players’ serving abilities  $f_{ij}, f_{ji}$ , we compute  $P_m(s_i, s_j, g_i, g_j, x_i, x_j)$  from every scoreline  $(s_i, s_j, g_i, g_j, x_i, x_j)$  in a match. To assess this model’s performance, we repeat this on every match in our dataset, with  $f_{ij}, f_{ji}$  computed with one’s method of choice.

### 4.3.1 Beta Experiments

The above approach only takes into account the current score and pre-calculated serve percentages when computing win probability. However, in many cases, relevant information may be collected from  $P_k$ . Consider the following in-match substring,

$P = \text{“}SSSS; RSSRRSS; SSSS; SRRSRSSSS; SSSS; RRRSSSRSSS;\text{”}$

The above sequence demonstrates a current scoreline of three games all. However,  $p_i$  has won 12/12 service points, while  $p_j$  has won 18/30 service points. If both players continue serving at similar rates,  $p_i$  is much more likely to break serve and win the match. Since original forecasts are  $f_{ij}, f_{ji}$  are based on historical serving percentages, it makes sense that in-match serving percentages may help us better determine each player’s serving ability on a given day. To do

this, we can update  $f_{ij}, f_{ji}$  at time  $t$  of the match to factor in each player's serving performance thus far.

Bevc explored this method by modeling each player's serving percentage  $f_{ij}$  with a beta prior  $b_{prior}$  that could be updated mid-match to yield posterior estimates. Through beta-binomial conjugacy, we can update  $b_{prior}$  when a player has won  $s_{won}$  points on serve out of  $s_{total}$  trials.

$$b_{prior} \sim \text{Beta}(a, b)$$

$$b_{posterior} \sim \text{Beta}(a + s_{won}, b + s_{total} - s_{won})$$

If  $f_{ij}$  is our prior,  $a, b$  must satisfy  $E(b_{prior}) = \frac{a}{a+b} = f_{ij}$ . Then  $a$  is a hyperparameter that determines the strength of our prior. To find a suitable prior strength, we will test various values of  $a$  via cross-validation on our training set. Regardless of  $a$ , the match's influence on our posterior serve estimates always grows as more points have been played. At any point in the match, we can always obtain a posterior serving percentage for either player of the form

$$f'_{ij} = E(b_{posterior}) = \frac{a + s_{won}}{a + b + s_{total}}$$

### 4.3.2 Elo-induced Serve Probabilities

Earlier on, Klaassen and Magnus suggested a method to infer serving probabilities from a pre-match win forecast  $\pi_{ij}$ . By imposing a constraint  $f_{ij} + f_{ji} = t$ , we can then create a one-to-one function  $S : S(\pi_{ij}, t) \rightarrow (f_i, f_j)$ , which generates serving probabilities  $\hat{f}_{ij}, \hat{f}_{ji}$  for both players such that  $P_m(0, 0, 0, 0, 0, 0) = \pi_{ij}$ <sup>1</sup>. As this paper was published in 2002, Klaassen and Magnus inverted their match probability equations to produce serve probabilities for ATP rank-based forecasts. However, since elo outperforms ATP rank, we apply this method to elo forecasts.

Due to branching complexity (see section 2.5), our hierarchical Markov model's match probability equation has no analytical solution to its inverse, even when we specify  $f_{ij} + f_{ji} = t$ . Therefore, we turn to the following approximation algorithm to generate serving percentages that correspond to a win probability within  $\epsilon$  of our elo forecast's:

---

#### Algorithm 1 elo-induced serve probabilities

---

```

procedure ELOINDUCEDSERVE(PROB, SUM,  $\epsilon$ )
  s0  $\leftarrow$  sum/2
  currentProb  $\leftarrow$  .5
  diff  $\leftarrow$  sum/4
  while |currentProb - prob|  $> \epsilon$  do:
    if currentProb < prob then
      s0 += diff
    else
      s0 -= diff
    diff = diff/2
    currentProb  $\leftarrow$  matchProb(s0, sum-s0)
  return s0, sum-s0

```

---

<sup>1</sup> $f_{ij}$  and  $f_{ji}$  are computed as specified in 3.4 with James-Stein normalization, so as to prevent extreme results

To generate elo-induced serve probabilities for a given match, we run the above algorithm with  $\text{PROB}=\pi_{ij}$ ,  $\text{SUM}=f_{ij} + f_{ji}$ , and  $\epsilon$  set to a desired precision level <sup>2</sup>. At each step, we call `matchProb()` to compute the win probability from the start of the match if  $p_i$  and  $p_j$  had serve probabilities  $f_{ij} = s0$ ,  $f_{ji} = \text{sum} - s0$ , respectively. Then we compare `currentProb` to `prob` and increment `s0` by `diff`, which halves at every iteration. This process continues until the serve probabilities `s0`, `sum-s0` produce a win probability within  $\epsilon$  of  $\text{PROB}$ , taking  $O(\log \frac{1}{\epsilon})$  calls to `matchProb`.

This inverse algorithm is useful for several reasons. Given any effective pre-match forecast  $\pi_{ij}$ , we can produce serve probabilities that are consistent with  $\pi_{ij}$  according to our hierarchical Markov model. By setting the constraint  $f_{ij} + f_{ji} = t$ , we also ensure that the sum of our players' serve probabilities agrees with historical data. While Klaassen and Magnus argue that  $t = f_{ij} + f_{ji}$  is largely independent of  $\pi_{ij}$ ,  $t$  holds greater importance when predicting specific score lines a match [3]. Using the hierarchical Markov Model's equations, Barnette specifically computed the probability of reaching any set score given  $f_{ij}, f_{ji}$ . When comparing probabilities across matches, one can observe that as  $t$  increases, closer set scores and tie-break games become more likely <sup>3</sup>. That is,  $t$  encodes information regarding likely trajectories of a match scoreline and relative importance of winning each game on serve. For higher  $t$ , service games are won more often and service breaks hold relatively more importance, while the opposite holds for lower  $t$ . Now, given  $\pi_{ij}$  and  $t$ , we can produce elo-induced serve probabilities for any two players.

### 4.3.3 Significance of $f_{ij} + f_{ji}$

To illustrate the importance of  $t = f_{ij} + f_{ji}$ , we consider the two following matches:

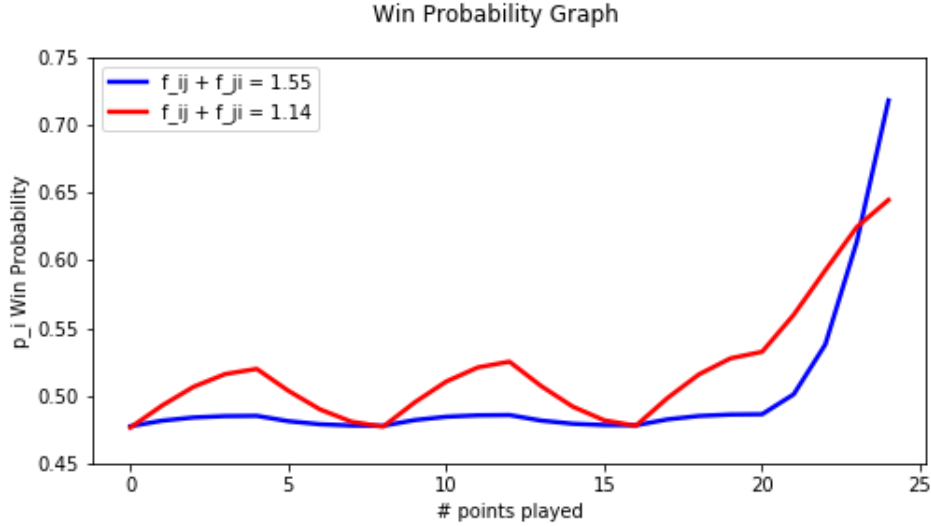
match	$p_i$	$p_j$	tny_name	surface	$\pi_{ij}$	$t$
$m_1$	Feliciano Lopez	John Isner	ATP Shanghai 2014	hard	.478	1.55
$m_2$	Andreas Seppi	Juan Monaco	ATP Kitzbühel 2014	clay	.476	1.14

In both matches,  $\pi_{ij}$  is approximately the same while  $t$  differs significantly. To visualize how service breaks can have varying importance, consider win-probability graphs corresponding to the following progression of points,  $P$ . In each scenario,  $p_i$  and  $p_j$ 's serve probabilities are calculated from method 4.3.2.

$P = \text{"SSSS;SSSS;SSSS;SSSS;SSSS;RRRR"}$

<sup>2</sup>for purposes of this project, setting *epsilon*=.001 is sufficiently accurate

<sup>3</sup>"closer" scores meaning 7-6, 6-7, 7-5, 5-7, etc.



Over the first five games, both players win every point on serve. When  $t = 1.55$ , service holds are expected and win probability hardly moves. After  $p_i$  breaks serve in the sixth game, his win probability suddenly spikes up, indicating a substantial advantage. When  $t = 1.14$ , our model assumes that players win service games with much lower probability. Therefore, each time a player holds serve, his win probability noticeably increases. When  $p_i$  breaks in the sixth game, his win probability climbs, but not as dramatically as in the previous case. When service games are won at a lower probability, service break advantages become weaker.

$t$ 's influence on the trajectory of a match suggests that prediction from any in-match scoreline will be most effective when  $t$  accurately reflects the players' overall serving ability. While elo ratings or betting odds yield the most effective pre-match predictions [9], their implied odds offer no information about  $t$  in a given match-up. As  $t$  holds significance for in-match forecasting from specific scorelines, method 4.3.2 allows us to combine any effective pre-match forecast  $\pi_{ij}$  with  $t$  to produce new serving probabilities.

## 4.4 Results

The following models were trained on 4584 matches from 2011-2013 and tested on 1855 matches from 2014 in our point-by-point data set. For consistency's sake, we only used best-of-three ATP tour-level matches and excluded matches from the Davis Cup. From cross-validation on the training set, we determined that setting hyper-parameter  $a = 300$  yielded optimal performance across beta experiments. We then added a beta variation to KM elo-induced and KM logit-induced, our two best-performing point-based models.

**LR** - Logistic Regression model with feature set specified in 4.1

**Random Forest** - Random Forest model with feature set specified in 4.2

**Equivalent** - KM model with serve probabilities set to tour-level averages (assumes equal player ability)

**elo-induced** - KM model with serve probabilities generated from elo-based win probability

**logit-induced** - KM model with serve probabilities generated from logistic regression on elo and surface-based elo

method	accuracy	log loss	calibration
LR 1)	71.4	.540	.993
LR 2)	75.2	.505	1.006
LR 3)	76.2	.488	.993
Random Forest	70.1	.562	.902
KM Equivalent	71.5	.538	1.040
KM James-Stein	74.9	.501	1.002
KM adjusted James-Stein	75.9	.498	.992
KM elo-induced	76.2	.486	1.001
KM elo-induced (a=300)	76.3	.481	1.003
KM logit-induced	76.5	.481	.996
KM logit-induced (a=300)	76.5	.477	.999

Of our models, Random Forest had the least predictive power. While this is perhaps due to difficulty in constructing an effective feature space, nearly all features fed into each machine learning model had an additive effect on win probability. Positive score differentials, break advantages, and elo differences will all boost a player’s expected win probability. The only non-additive features we considered were “set” and “surface”. While the random forest model was able to consider various situations among these two features, we conclude that its inclination to group similar examples together, rather than estimate linear relationships between variables, ultimately accounted for its performance.

Logistic regression, on the other hand, is designed to learn additive relationships between features and its probability output. It’s performance with the full feature set was competitive with our point-based models, despite their superior grasp of tennis’ scoring system <sup>4</sup>. Presumably, this model holds an advantage over point-based models in its ability to fit itself to the training data. Aside from our hyper-parameter search of  $a$  in beta experiments, the point-based models offer no comparable method to optimize performance with respect to a training set.

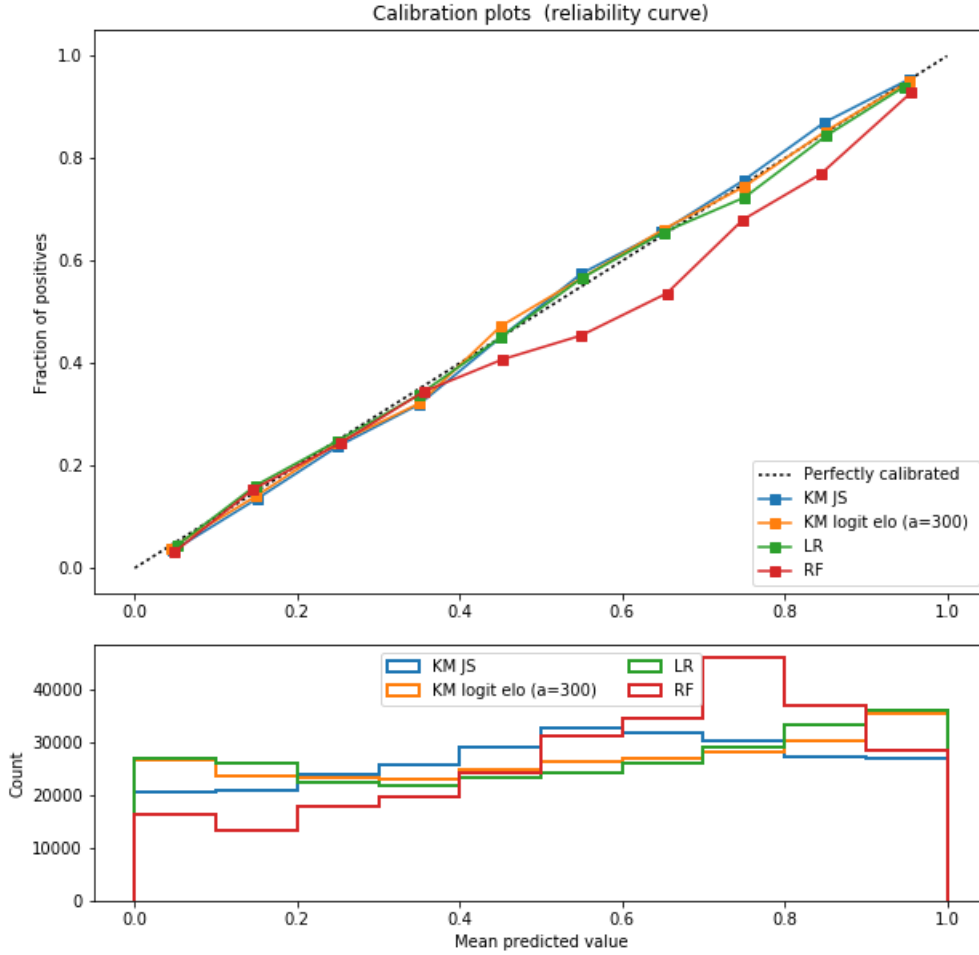
Of the point-based models, KM logit-induced (a=300) achieved the best performance, although the improvement from running beta experiments with a=300 was marginal. Since elo ratings produce more reliable pre-match win forecasts  $\pi_{ij}$  than estimates of  $f_{ij}, f_{ji}$  from historical serve/return data, using serve probabilities induced from elo ratings and then from our logit elo model improved our models performance by significant increments. Over the next few sections, we will examine our models’ calibration and performance across subsets of our dataset.

---

<sup>4</sup>consider, for example, how our logistic regression has no idea what a tiebreak game is



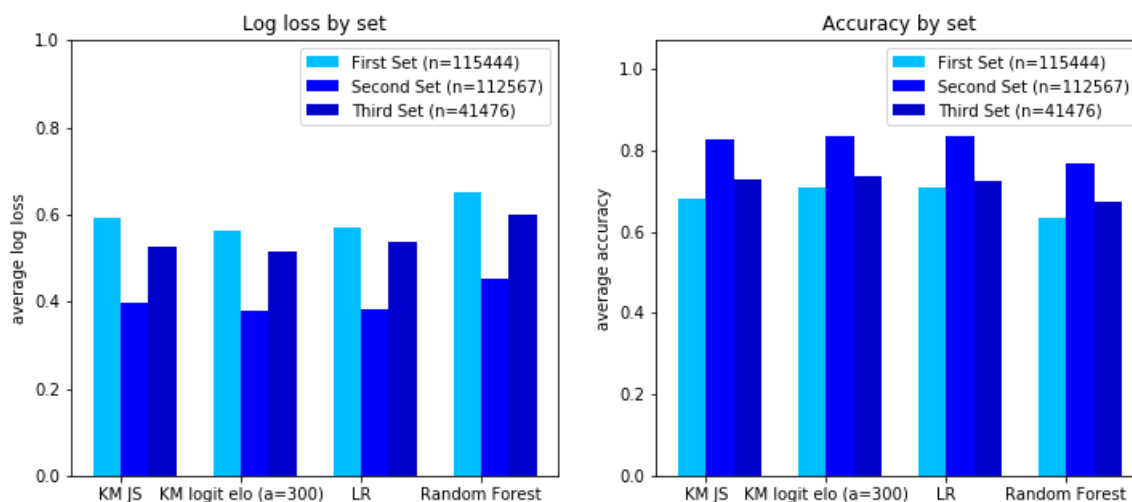
#### 4.4.1 Model Calibration



Calibration represents the ratio between observed rate of success and mean predicted rate of success. From a perfectly calibrated model that predicts a win probability of  $p$ , we can always expect a win to occur with probability  $p$ . While the Random Forest model under-predicted wins, the rest of our models closely resemble a perfectly calibrated model. From our table, KM logit elo and KM JS exhibited near-perfect calibration of 1.001 and .999, while logistic regression reported .993. As there is degree of uncertainty in measuring calibration, we could not reasonably expect to produce more well-calibrated models. Given win probability forecast  $\pi_{ij}$  at any point in the match, we expect  $p_i$  to win with probability  $\pi_{ij}$ .

## 4.4.2 By Set

to do: display uncertainty lines on each bar?? not sure if this is meaningful?



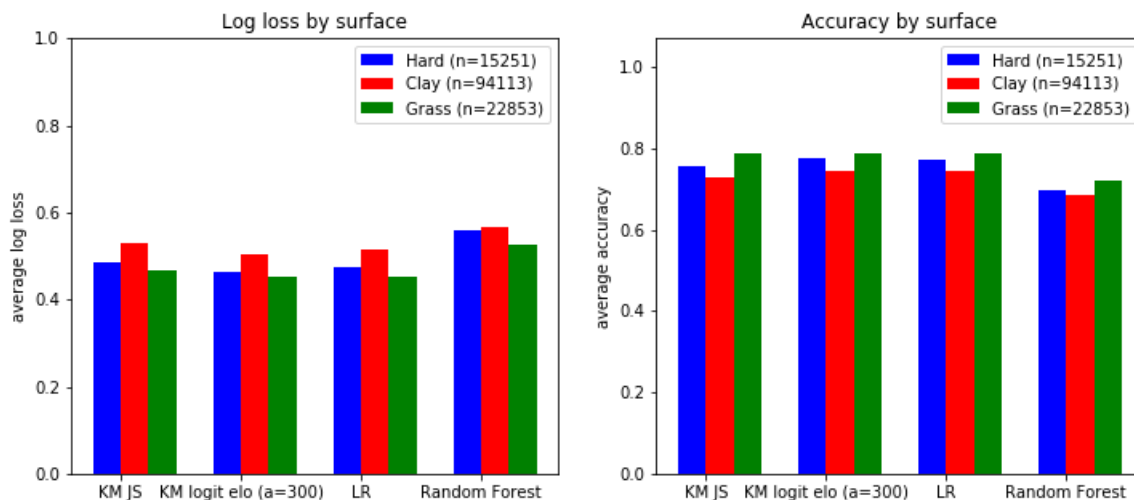
Since we have less information about a match early on, it is always most difficult to forecast matches during the first set. However, across both metrics, models are most effective during the second set. By this time in the match, the winner of the first set is often a clear favorite to win the match, unless their opponent has a significantly higher elo rating or serve percentage  $f_{ji}$ . As 64.3% of matches in our test set are straight-sets victories <sup>5</sup>, many matches offer relatively easy predictions for our model during the second set.

Uncertainty increases once the match enters a third set because the set score has evened at 1-1. While the set score is tied in both first and third sets, it is easier to predict the outcome during the third set because its outcome corresponds directly to the match outcome. When predicting in the first set, the player who wins the first set can still go on to lose the match, which increases uncertainty.

---

<sup>5</sup>matches that end in two sets are straight-sets

### 4.4.3 By Surface



Zoom in on graphs (?)

By surface, models predicted grass court matches most effectively and clay court matches least effectively. By nature, grass courts are faster and conducive to offensive play. Since service games are held at a higher rate, break advantages can be more decisive, which likely gave our models an edge when predicting outcomes from all scorelines. On the other hand, clay courts suit defensive play and exhibit more service breaks. When service breaks are more likely, it becomes harder for our models to predict the match because advantages from a current score line are more likely to be turned around. Hard courts, where the majority of matches took place, ultimately proved in between grass and clay in predictability. Further insight into the volatility of matches <sup>6</sup> may formally shed light on differing results across surfaces.

### 4.4.4 Distribution of Performance

show a histogram of log loss scores, accuracy scores by match. Do we want this (??)

### 4.4.5 Variable Importance

Discuss relative variable importance among features in the most effective LR model. Talk about how these features relate to the point-based model or how there may be features with information the point-based model lacks (likely not, since I crafted features to represent scenarios which are embedded in the point-based model, like break point, etc). serve percentages could be analogous to beta experiments

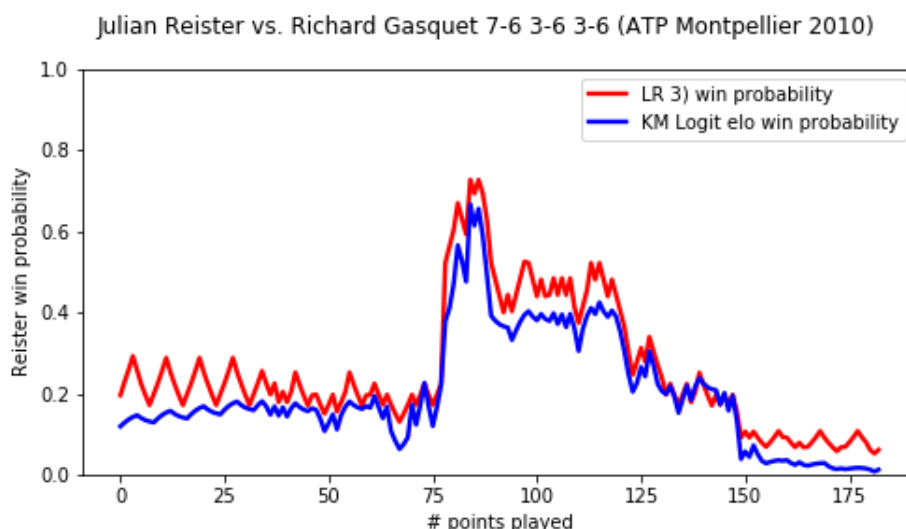
graph LR and KM together, commenting how LR jumps around more

<sup>6</sup>we may calculate a measure of volatility from point-by-point sequences or a model's win probability output

(from points and game increases, this is a matter of reactivity from correlation vs causality; lr is somewhat reacting to jumps because having a point lead is going to be more associated with winners.)

## 4.5 Visualizing Win Probability

As the most effective Logistic Regression and KM models were not too far off in predictive performance, we now visualize their forecasts with several tour-level matches.



Over the first thirty points of the match, the contrast between our models is reminiscent of 4.3.3. Since our logistic regression model has learned specific coefficients for the “point.diff” feature, Reister’s win probability spikes up and down noticeably as he and Gasquet exchange games on serve. Our KM logit elo model seems to have set  $t = f_{ij} + f_{ji}$  to a high value, expecting frequent service holds and producing a nearly static win probability over this time frame. While we can communicate the relative importance of service breaks with constraint  $t = f_{ij} + f_{ji}$ , there is no direct method for controlling probability fluctuations with logistic regression.

For the last thirty points of the match, consider how Reister’s win probability slowly declines according to the KM model, while it remains approximately constant according to logistic regression. One drawback of logistic regression is that it cannot distinguish between situations whose score differentials are equivalent. As Gasquet maintains a break advantage through the end of third set, this explains logistic regression’s behavior, as the score differentials remain similar.

In general, consider a player serving at (1,1,5,4,3,0) and one serving at (1,1,1,0,3,0). In the first situation, the player serving wins the match if he wins any of the next three points. From the second scenario, the player serving holds a break advantage early in the final set, from which the returner has many more chances to come back. Assuming each player serves at

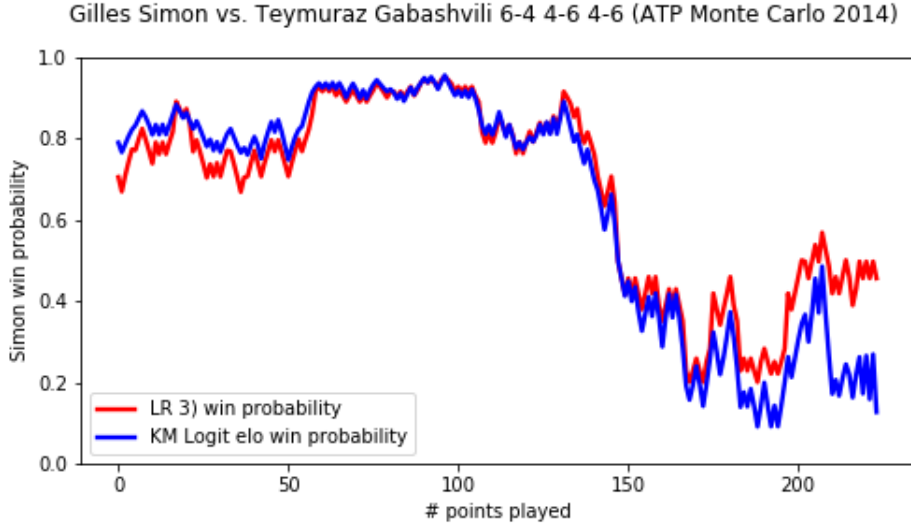
$f_i = f_j = .64$ , our win-probability equation suggests a substantial difference between these two scenarios:

$$P_m(1, 1, 5, 4, 3, 0) = .994$$

$$P_m(1, 1, 1, 0, 3, 0) = .800$$

Although the first situation is clearly favorable, logistic regression will observe equivalent differentials (set\_diff, game\_diff, and point\_diff) and compute approximately the same probability in both scenarios <sup>7</sup>.

In a similar example, logistic regression may also fail to notice when a higher-ranked player is about to lose a close match.



Simon leads for most of the match until Gabashvili comes back to force the third set. At the end of the third set, Gabashvili serves out the match from 5-4, yet the model fails to detect that Simon is about to lose. This is because, with sets at 1-1 and game differentials nearly even, Simon's elo advantage outweighs his current scoreline disadvantage. Consequentially, his win-probability hovers between 50% as he loses the match, rather than fall toward 0. Still, when comparing model performance in specific scenarios, it is important to note that the KM model was specifically designed for tennis prediction. Our logistic regression model, on the other hand, was simply fed a bunch of score-related features that seemed helpful. Despite apparent faults, its performance suggests potential in further exploring machine learning methods.

---

<sup>7</sup>after fitting coefficients for the equation  $P(\text{win}) = \text{logit}(s_i, s_j, g_i, g_j, x_i, x_j) = \frac{e^{(c_1 s_i + c_2 s_j + c_3 g_i + c_4 g_j + c_5 x_i + c_6 x_j)}}{1 + e^{(c_1 s_i + c_2 s_j + c_3 g_i + c_4 g_j + c_5 x_i + c_6 x_j)}}$ , coefficients  $c_1 \approx c_2, c_3 \approx c_4, c_5 \approx c_6$  by symmetry and therefore  $\text{logit}(1, 0, 5, 4, 3, 0) \approx \text{logit}(1, 0, 1, 0, 3, 0)$

## 5. Conclusion

For the first time, we have documented the performance of in-match prediction methods on thousands of matches. While previous work only applied methods to individual matches or the 1992-1995 Wimbledon match data set [?], we have applied methods to thousands of matches from the past several years. Results from this project and Jeff Sackmann’s point-by-point dataset should serve as an appropriate benchmark for future work in this field.

Although many papers have referenced Klaassen and Magnus’ point-based model, none has explicitly outlined a way to correct for uncertainty stemming from small sample sizes in players’ serve/return history. Applying a normalization method, such as the James-Stein estimator in 3.5, significantly boosts the performance of point-based models. By factoring in the overall ability of a player’s opponents in the past twelve months, calculating  $f_{ij}, f_{ji}$  from the opponent-adjusted method in 3.6 offers additional predictive power. Still, none of these variations produce pre-match forecasts as effectively as elo ratings. Using our approximation algorithm in 4.3.2, we can derive serve probabilities from any pre-match forecast  $\pi_{ij}$  and historical serve parameter  $t = f_{ij} + f_{ji}$ . A combination of this method, using  $\pi_{ij}$  derived from an elo-based logistic regression, ultimately produced the most reliable in-match prediction model.

Finally, logistic regression’s performance with a score-based feature set proved noteworthy. Machine-learning models offer potential in their ability to learn from a training set. While serve parameters  $f_{ij}, f_{ji}$  may be obtained from a machine-learning model’s forecast,  $\pi_{ij}$ , our traditional point-based hierarchical Markov Model offers no comparable method to “train” itself from a dataset. Future work with point-based models that offers some aspect of optimization on a training set may be of much interest to fans and bettors alike.

### 5.1 Real-World Applications

Methods explored in this project may be applied to betting markets in several ways. First, we can take any pre-match forecast,  $\pi_{ij}$ , derived from implied market odds and produce corresponding serve probabilities  $f_{ij}, f_{ji}$  from method 4.3.2. As Kovalchik found that the Bookmaker Consensus Model outperformed elo ratings [9], we could likely further improve performance of in-match prediction models by using pre-match betting odds. Next, we may back test any of our in-match prediction techniques against historical in-match odds. While Betfair offers historical data, all market data is time-stamped and lacks corresponding match scores. While mapping time-stamps to their corresponding score lines over thousands of matches proved outside the scope of this project, analyses with in-match betting data do exist, as when Huang

demonstrated score inference from betting odds [6]. Finally, one may also test betting strategies with any of these models in real time, provided you have up-to-date information to produce elo ratings and  $f_{ij}, f_{ji}$ .

There are many advances yet to come in forecasting in-game win probability of tennis matches. Aside from beta experiments, all point-based models in this paper operated under the assumption that points played by each server are iid. While Klaassen and Magnus concluded that deviations from iid are small enough for such models to provide a reasonable approximation [7], there is clearly much more work to be done in this area. Tracking a player’s momentum at any point in the match, and updating serve probabilities accordingly, may also allow us to improve our models.

## 5.2 Future Steps

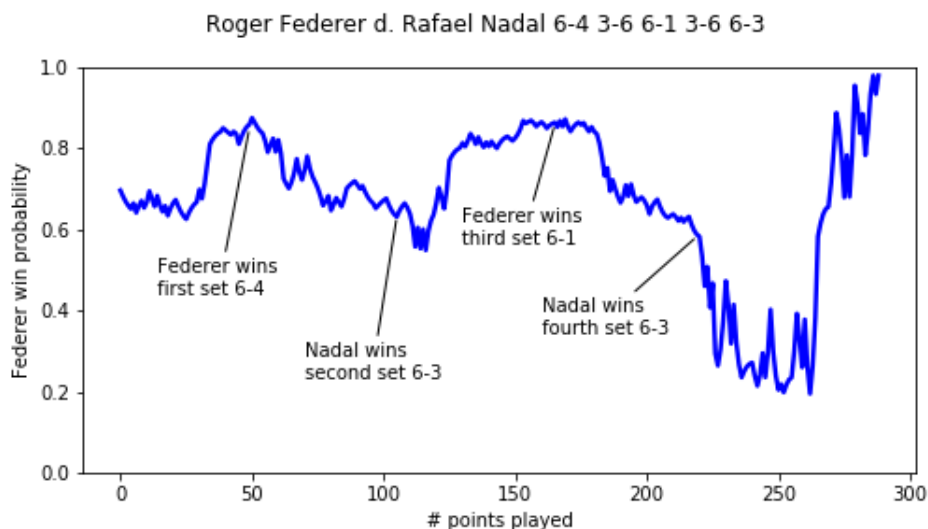
apply to betting markets

apply to women’s matches

## 5.3 Visualizing Grand Slam Matches

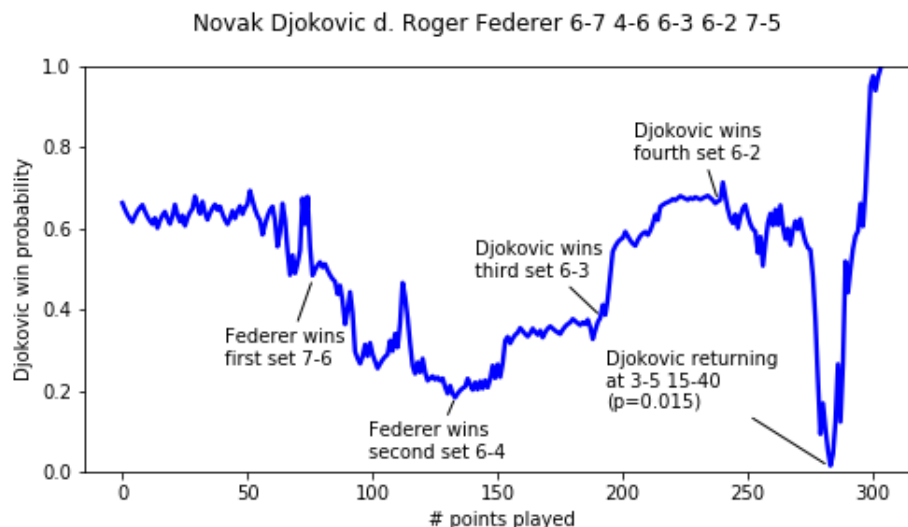
Earlier this year, Roger Federer overcame Rafael Nadal in the Australian Open final 6-4 3-6 6-1 3-6 6-3. Below, we view a win-probability graph corresponding to our LR 3) and KM logit-induced ( $\alpha=300$ ) models.

### 5.3.1 Roger Federer vs. Rafael Nadal Australian Open 2017 Final



### 5.3.2 Novak Djokovic vs. Roger Federer US Open 2011 Semi-Final

One of the most famous comebacks in tennis history. Djokovic recovered from a two-set deficit before escaping defeat at 3-5 15-40 on Federer's serve in the final set <sup>1</sup>.



<sup>1</sup>if one were to consider Djokovic's decision to hit an all-or-nothing return winner at 3-5 15-40 in the fifth set, we could have computed his win probability to be even less than  $p = .015$ , as Federer would have surmised <https://www.theguardian.com/sport/2011/sep/11/us-open-2011-federer-djokovic>. Calculating in-match win probability within points based on shot-selection is realistic nowadays, thanks to Jeff Sackmann's match charting project and recent data analyses with Hawkeye ball-tracking data, and a logical extension to our point-by-point models



# Bibliography

- [1] Tristan Barnett and Stephen R Clarke. Combining player statistics to predict outcomes of tennis matches. *IMA Journal of Management Mathematics*, 16(2):113–120, 2005.
- [2] Tristan J Barnett, Stephen R Clarke, et al. Using microsoft excel to model a tennis match. In *6th Conference on Mathematics and Computers in Sport*, pages 63–68. Queensland, Australia: Bond University, 2002.
- [3] Tristan J Barnett et al. *Mathematical modelling in hierarchical games with specific reference to tennis*. PhD thesis, Swinburne University of Technology, 2006.
- [4] Martin Bevc. Predicting the outcome of tennis matches from point-by-point data. 2015.
- [5] Bradley Efron and Carl N Morris. *Stein's paradox in statistics*. WH Freeman, 1977.
- [6] Xinzhuo Huang, William Knottenbelt, and Jeremy Bradley. Inferring tennis match progress from in-play betting odds. *Final year project*, Imperial College London, South Kensington Campus, London, SW7 2AZ, 2011.
- [7] Franc J G M Klaassen and Jan R Magnus. Are points in tennis independent and identically distributed? evidence from a dynamic binary panel data model. *Journal of the American Statistical Association*, 96(454):500–509, 2001.
- [8] Franc JGM Klaassen and Jan R Magnus. Forecasting the winner of a tennis match. *European Journal of Operational Research*, 148(2):257–267, 2003.
- [9] Stephanie Ann Kovalchik. Searching for the goat of tennis win prediction. *Journal of Quantitative Analysis in Sports*, 12(3):127–138, 2016.
- [10] Michal Sipko and William Knottenbelt. Machine learning for the prediction of professional tennis matches. 2015.