Contents

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. 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 one team or player is favored to win. While one can produce this from any model of choice, the field strives to produce the most well-informed estimate of how likely a player is to win the match. With the recent proliferation of online betting, in-match win probability dictates an entire market of its own. While tennis has yet to broadcast win probabilities, Betfair reportedly traded over 40 million in volume over the 2015 Wimbledon Final (footnote). Furthermore, around 80% of such betting occurs while matches are in progress (Sipko). Clearly, in-match win probability is of great concern to all participants in this betting market.

1.2 History of Tennis Forecasting

Plenty of research on tennis match prediction exists over the past twenty years. In (are points in tennis iid?), Klaassen and Magnus test the assumption that points in a tennis match are independent and identically distributed. 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 players probability of winning a point on serve (forecasting). Barnett and Clarke then offer a method of estimating each players serve probability from historical data (combining player outcomes). Years later, Bevc proposed updating each players serve probability with a beta distribution between each point and computing the corresponding win probability with the above model (predicting the outcome). Recently, Kovalchik assessed performance of 11 different pre-match prediction models on all tour-level ATP matches in 2014.

Over the past several years, Jeff Sackmann (offer footnote to website) 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 matchesboth tour-level and satellite–and a crowd-sourced match-charting project spanning over 3000 matches, where volunteers record each shots type and direction over an entire match. While 538 and Kovalchik use Jeff Sackmanns match data generate elo ratings, none of the aforementioned papers have used his point-by-point dataset.

As these papers have spanned several decades, the datasets referenced among them are not consistent. ", : both test in-play models on around 500 matches from Wimbledon in 1991-94. Barnett focuses almost exclusively on a marathon match between Andy Roddick and " El Aynoui from the 2002 Australian Open.

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 several new concepts to these datasets, from successful probability models in football and baseball, to state exploration via hidden Markov Models.

(Can reference older sports forecasting mentioned in Nettleton and Lock...)

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 match's progression. Each match contains a string listing the order in which players won points and switched serve. As the ATP 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 [1]. Due to observed inconsistencies, we employed canonicalization of player names between match and point-by-point datasets [2] in order to maximize the number of available matches with point-by-point summaries. As a portion of the point-by-point summaries are from satellite events, we were able to match around 10,600 tour-level matches. Once this was achieved, we could then associate information generated from the match dataset's entirety (elo ratings, year-long adjusted serve stats, etc) with these 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`. Implementations of each method in this project, and instructions on how to generate all relevant statistics, are also provided.

[1] eg. hash(matchX) = "Roger Federer Tomas Berdych 2012 3-6 7-5 7-5"
[2] 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 player k's score in sets, games, and points, respectively. 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) [1].

The majority of in-play tennis models utilize a hierarchical Markov Model, which embodies the levels in tennis' scoring system. Barnett formally defines a representation for scores in tennis (Barnett Clarke 2002). 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 *iid*, we can then use the below model to recursively determine win probability:

$P_m(s_i, s_j, g_i, g_j, x_i, x_j) =$ probability that p_i wins the match when serving from this scoreline

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

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 [2]. 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 $P_g(3,3) = \frac{f_{ij}^2}{f_{ij}^2 + (1 - f_{ij})^2}$.

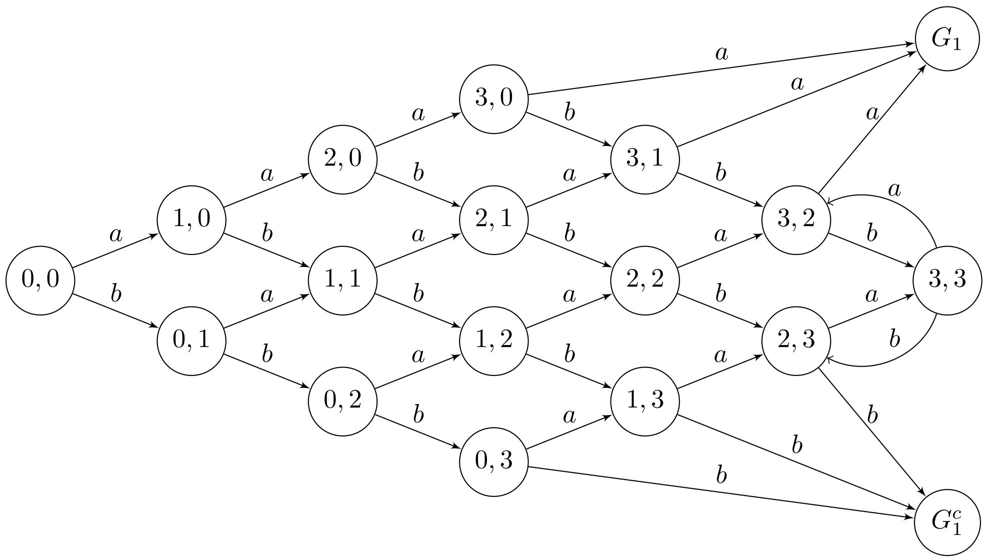
Possible sequences of point scores in a game:

a - player i wins the following point

[1] The best-of-five format is typically reserved for men's grand slam and Davis Cup events

[2] 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

b - player j wins the following point



Boundary values:

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

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.

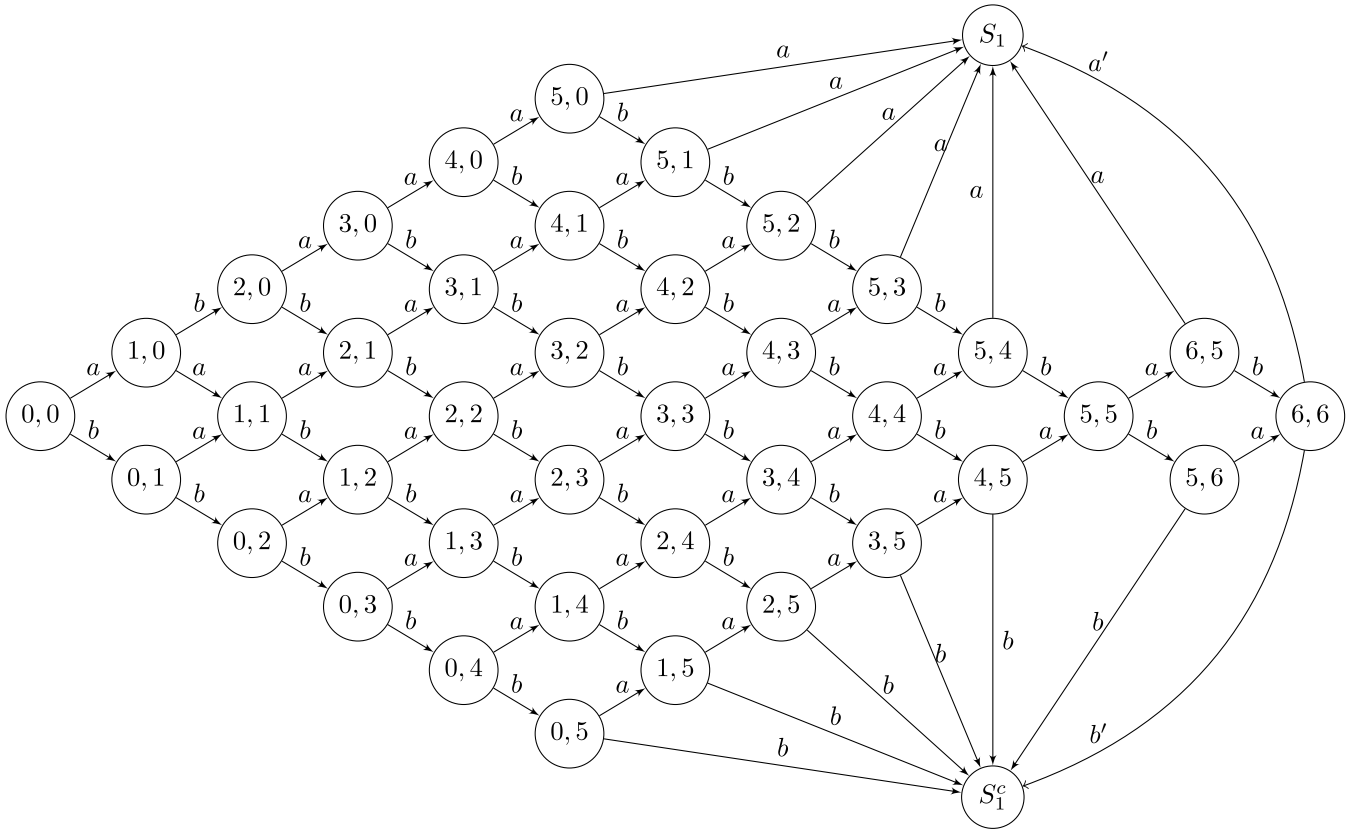
Possible sequences of point scores in a game:

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:

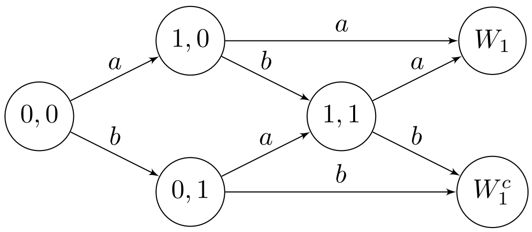
$$P_s(g_1, g_2) \begin{cases} 1, & \text{if } g_1 \geq 6, g_1 - g_2 \geq 2 \\ 0, & \text{if } g_2 \geq 6, g_2 - g_1 \geq 2 \\ P_{tb}(s_1, s_2), & \text{if } g_1 = g_2 = 6 \\ P_g(0,0)(1 - P_s(g_2, g_1 + 1)) + (1 - P_g(0,0))(1 - P_s(g_2 + 1, g_1)), & \text{otherwise} \end{cases} \tag{2.2}$$

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

2.3 Modeling a best-of-three match

a - player 1 wins the following set

b - player 2 wins the following set



Boundary values:

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

Combining the above equations, we can recursively calculate win probability with player i serving from $(s_i, s_j, g_i, g_j, x_i, x_j)$ as:

$P_m(s_i, s_j, g_i, g_j, x_i, x_j) = f_{ij} * P_m(s_i, s_j, g_i, g_j, x_i+1, x_j) + (1 - f_{ij}) P_m(s_i, s_j, g_i, g_j, x_i, x_j+1)$
$= P_g(x_i, x_j) * (1 - P_m(s_j, s_i, g_j, g_i+1, 0, 0)) + (1 - P_g(x_i, x_j)) * (1 - P_m(s_j, s_i, g_j+1, g_i, 0, 0))$
$= P_g(x_i, x_j) * (1 - (P_s(g_j, g_i+1) * P_m(s_j+1, s_i) + (1 - P_s(g_j, g_i+1)) * P_m(s_j, s_i+1)) + (1 -$
$P_g(x_i, x_j)) * (1 - (P_s(g_j+1, g_i) * P_m(s_j+1, s_i) + (1 - P_s(g_j+1, g_i)) * P_m(s_j, s_i+1)) = ...$

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 player i and j's entire match histories, while the BCM model incorporates all information encoded in the betting market. As we will later back-test effective in-match models against market data, we first explore all pre-match prediction methods that do not employ market data.

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 match 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 538's analysts, elo ratings perform optimally when allowing K_{it} to decay slowly over time (How We're Predicting). With $m_i(t)$ representing the p_i's career matches played at time t we update our learning rate:

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

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 initalize 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 [1] in chronological order, storing $E_i(t), E_j(t)$ for each match and updating each player's elo accordingly.

3.3 ATP/WTA Rank

While Klaassen and Magnus incorporated ATP rank into their prediction model (forecasting 2003), Kovalchik and 538 concur that elo outperforms ranking-based methods. On ATP match data from 2010-present, we found:

Table with elo vs ATP/WTA rank

Considering their superiority to ATP rank in 21st-century matches, models in this paper use elo ratings to represent a player's ability.

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}. Klaassen and Magnus outline a way to estimate each player's serving probability from historical serve and return data.

$$f_{ij} = f_t + (f_i - f_{av}) - (g_j - g_{av})$$
$$f_{ji} = f_t + (f_j - f_{av}) - (g_i - g_{av})$$

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 player i and j's percentage of points won on serve and return, respectively. f_{av}, g_{av} are tour-level averages in serve and return percentage. Since all points are won by either server or returner, $f_{av} = 1 - g_{av}$.

As per Klaassen and Magnus' implementation, 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 played at tournament w in year y.

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

Klaassen and Magnus 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 to estimating players' abilities.

Implementing this method with year-to-date statistics proves troublesome because f_i, g_i decrease significantly in uncertainty as player i accumulates matches throughout the year. Due

[1] tennis' Open Era began in 1968, when professionals were allowed to enter grand slam tournaments. Before then, only amateurs played these events

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 [2]:

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

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 $M_p(0,0,0,0,0,0) = .5139$

3.5 James-Stein Estimator

Decades ago, Efron and Morris described a method to estimate groups of sample means (Efron Morris 1977). The James-Stein estimator shrinks sample means toward the overall mean, in proportion to its estimator's variance. Regardless of the value of θ, this method has proven superior to the MLE method (reporting the sample mean for each group), 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.

[2]for the current month m, we only collect month-to-date matches

To shrink serving percentages, we compute the variance of all recorded f_i statistics [3] 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 held only one tour-level match in his year-long stats. From a previous one-sided victory, his 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 estimate that Elahi has an 89.3% chance of winning points on serve. 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 Elahi's victory, despite only having collected his player statistics for one match. Karlovic's 360-point elo advantage, which calculates Elahi's win probability as $\hat{\pi}_{ij}(t) = (1 + 10^{\frac{1876.9545-1516.9178}{400}})^{-1} = .1459$, leads us to further questions 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.

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

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

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

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

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

[3] each f_i is computed from the previous twelve months of player data

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

given $JS(f_i), JS(f_j) : M_p(0,0,0,0,0,0) = .4806$

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. By shrinking the serve/return statistics, our model lower's Elahi's inflated serve percentage and becomes less vulnerable to small sample sizes.

Since overly confident forecasts can hurt model performance with respect to cross entropy, the James-Stein estimator allows a safer way to estimate outcomes of matches with lesser-known players. 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 statistics.

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:

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

$g_{i\_opp\_av}$ 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.

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

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 Original Methods

First we observe performance of elo variants and Klaassen-Magnus point-based models. Since their original implementations provide explicit formulas with no optimization component, we directly assess their performance on 2014 tour-level match data.

method	accuracy	log loss		
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)	69.4	.578		

By combining elo and surface elo, we achieve a log loss of .58. Aside from the Bookmaker Consensus Model, which draws information directly from the betting market, no other model is documented as doing this well. Earlier this year, Kovalchik reported 70% accuracy using 538's elo method (footnote). As one can observe online (https://github.com/skoval/deuce), she calculated elo ratings using all satellite events (futures and challengers) in addition to tour-level events. While this accounted for a one-percent increase in accuracy, her method achieved a comparable log loss of .59 (2017). For simplicity's sake, we will calculate all player ratings and statistics from tour-level matches alone.

(move to end) Aside from market odds-based models, no published approaches have reported log loss superior to .58 with recent tour-level matches. While Sipko claimed to have achieved 4.3% ROI off the betting market with a neural net, the best of his machine learning models achieves a log loss of .61 (2014). As Sipko surveyed logistic regression, the common-opponent model, and an artificial neural net, we are confident that elo provides a confident starting place for in-match prediction models.

3.7.2 Methods with Optimization Component

The following methods were trained on 2010-2013 match data and tested on 2014 ATP match data.

input	train accuracy	test accuracy	train log loss	test log loss
elo		69.2		.587
surface elo		68.6		.590
elo 538		69.3		.595
surface elo 538		69.7		.595
elo/surface elo				
KM				
KM James-Stein				
KM Surface				
KM Surface James-Stein				
KM adjusted				
KM adjusted James-Stein				

It is important to note that Klaassen and Magnus' method of combining player statistics involves no optimization with respect to a training dataset. Of the above methods, only a logistic regression with elo and surface elo actually learns its model parameters with respect to a training dataset.

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

P="SSRSS;RRRR;SRSSS;SRRSRSSS;SRSSRS;RSRSSS;SRSRSS;RSRSRSSS;SSSS.SSSRRRSS;

RSSSS;SSRSS;SSSRS;SSSS;RRRSSSSRRSSRRSRSSS;SRSRSS;SSSRS;RSRSSRSS;SSSS;SRSSS;

RSRSSRRSSS;R/SR/SS/RR/RS/SR.RSRRR;..."

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 $\{P_0, P_1, ..., P_{n-1}\}$ from a match with n total points, with P_i representing the subset of the match after i points have been played.

$P_0 =$ ""

$P_1 =$ "S"

$P_2 =$ "SS"

$P_3 =$ "SSR"

...

With $M = \{M_1, M_2, ...M_k\}$ complete match-strings in our point-by-point data set, the size of our enumerated data set then becomes $\sum_{i=1}^{k} |M_i|$. This comes out to 1231122 points for ATP matches and "" for WTA matches.

As many in-match prediction models utilize the hierarchical Markov Model structure, we may carry over previously computed serving percentages to in-match prediction. To start, we will test several machine-learning methods as a baseline.

4.1 Logistic Regression

Consider a logistic regression model.

$$logit$$

From any scoreline $(s_i, s_j, g_i, g_j, x_i, x_j)$, we can simply feed these values as parameters into our model. Logistic Regression's structure makes it easy to consider additional features for each player, such as elo difference, surface elo difference, etc. Before adding more features to the model, we consider two baselines: a model using $(s_i, s_j, g_i, g_j, x_i, x_j)$ and another model trained on elo differences and a lead heuristic L_{ij}.

This heuristic simply calculates one player'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 for prediction:

(table)

Next, we test the following combinations of features:

1) sets + games + points

2) lead-margin + elo diff + surface elo diff

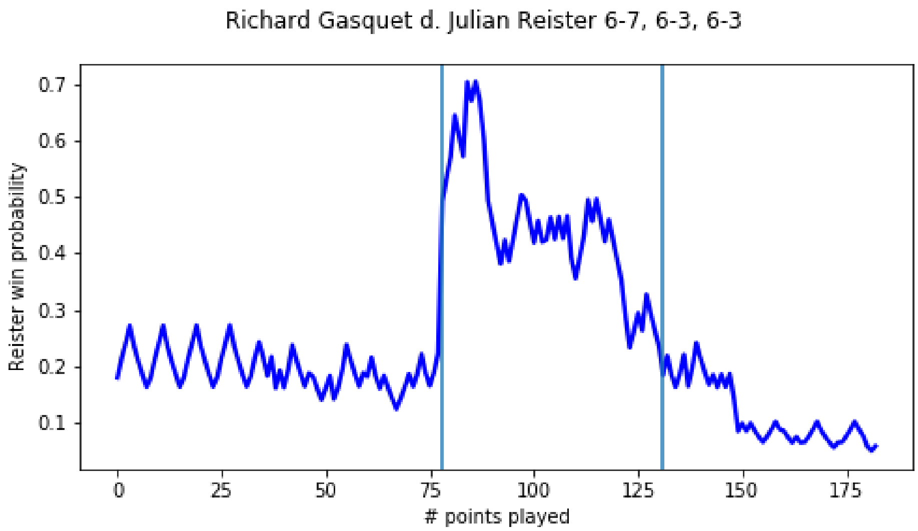
3) all features

4) Specific "score" features

4.1.1 Cross Validation

Each match in our best-of-three dataset has around 160 points on average. We implement five-fold group validation, keeping matches together, so points from the same match do not overlap between train, validation, and test sets. This prevents a single match from informing the model before its later assessed by the model. (need to get your datasets straight, best-of-three, best-of-five, men's, women's)

4.1.2 Visualizing Logistic Regression



One drawback of logistic regression is that it cannot distinguish between situations whose score differentials are equivalent. A player serving at (1,1),(5,4),(3,0) will have approximately the same win probability as one serving at (1,1),(1,0),(3,0). However, 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 only holds a break advantage early in the 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:

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

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

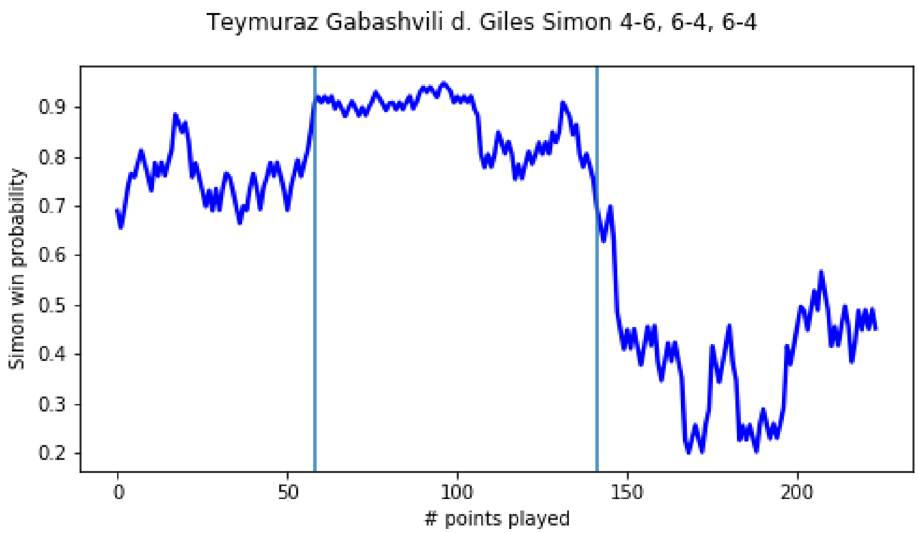
Although the first situation is clearly favorable, logistic regression will compute approximately the same probability in both scenarios [1]

Another issue is that logistic regression can fail to detect when a higher-ranked player is about to lose in a close match. Below,

[1] after fitting coefficients for the equation P(win) = 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 $logit(1, 0, 5, 4, 3, 0) \approx logit(1, 0, 1, 0dddd, 3, 0)$

Table 4.1: My caption

Variable	Description
$surface$	hard, clay, grass
set	first, second, third
$eloDiff$	Elo(p_0) - Elo(p_1)
$setDiff$	SetsWon(p_0) - SetsWon(p_1)
$breakAdv$	
$pointDiff$	
$brkPointAdv$	



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 similar scenarios and smoothing 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. Nettleton and Lock also deviate from the traditional random forest classification problem in using regression trees and averaging their estimates to produce a probability estimate, rather than a majority vote.

Then, training on our validation set, we test two random forest models on our point-by-point dataset, one with classification trees and one with regression trees.

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, testing all estimates of f_{ij}, f_{ji} (James-Stein normalized, player-adjusted, elo-induced, surface-specific)

4.3.1 Beta Experiments

The above approaches only take into account the current score when computing win probability. However, in many cases, there is much more information that may be collected from P_k. Consider the following in-match substring,

$P =$ "$SSSS; RSSSRRSS; SSSS; SRRSRSRSSS; SSSS; RRRSSSRSRSSS;$"

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 in the match.

"" attempted this method with beta experiments. The beta distribution is a generalization of the uniform distribution. We often use the beta distribution to represent prior and posterior estimates to some probability parameter b_{prior}.

To update our matches with in-match serving statistics, we set f_{ij} as a prior and update with the number of points won and played on $p_i's$ serve, (s_{won}, s_{pt}). Through beta-binomial conjugacy, we then obtain an update of the form

$b_{posterior} = \frac{\alpha * f_{ij} + s_{won}}{\alpha * f_{ij} + s_{pt}}$

where α is a hyper parameter that determines the strength of our prior. Regardless of alpha, the match's influence on our posterior serve estimates will always grow as more points have been played.

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 π_{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 $P_m(0, 0, 0, 0, 0, 0) = \pi_{ij}$. 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 past equation), 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 with ϵ of our elo forecast's:

Algorithm 1 elo induced serve probabilities

```
procedure EloInducedServe(prob,sum,ϵ)
    s0 ← sum/2
    currentProb ← .5
    diff ← sum/4
    while |currentProb - prob|>ϵ do:
        currentProb ← matchProb(s0,sum-s0)
        if currentProb <prob then
            s0 += diff
        else
            s0 -= diff
        diff = diff/2
    return s0,sum-s0
```

To generate elo-induced serve probabilities for a given match, we run the above algorithm with PROB=π_{ij}, SUM=$f_{ij} + f_{ji}$, and ϵ set to a desired precision level [2]. 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 s0,sum-s0, respectively. Then we compare currentProb to prob and increment by diff, which becomes half as big at every iteration. This process continues until the serve probabilities s0,sum-s0 produce a win probability within ϵ of PROB, with $O(\log \frac{1}{\epsilon})$ calls to matchProb.

This inverse algorithm is very useful for several reasons. Given a most effective pre-match forecast π_{ij}, we can produce serve probabilities for each player that are consistent with π_{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 π_{ij}, t is of greater importance when predicting likely scores of a match (reference to Madurska or Barnette or whoever did that). That is, t encodes information regarding likely trajectories of a match scoreline and relative importance of service breaks. Examples: Federer vs Isner at Paris, then Ferrer vs. Nishikori/Schwartzman or something with really good returners.

t's influence on the trajectory of a match suggests that prediction from any in-match scoreline will be most effective if serve probabilities are consistent with t.

4.4 Results

[2] for most purposes, setting *epsilon*=.001 is sufficiently accurate