University of Nottingham
UK | CHINA | MALAYSIA

Department of Mathematical Sciences

MATH4045 - Statistical Data Analysis and Modelling

Ranking system for ATP Tennis Matches

Authors:

Piratheesh Raveenthira (pmypr3@nottingham.ac.uk)

Nikhil Passi (pmynp7@nottingham.ac.uk)

Kelly Nguyen (pmykn1@nottingham.ac.uk)

Shiyang Li (smysl4@nottingham.ac.uk)

Yichi Zhang (smyyz6@nottingham.ac.uk)

Date: 28th October 2023

Abstract

This report aims to devise and evaluate models that rank male tennis players: the Bradley-Terry model and the Elo rating system, based on ATP data from 2000 to 2020. The basic constructions of these 2 models are first introduced, and then refined with the consideration of different tournaments and match characteristics. The models were validated using Kendall and Spearman's correlation coefficients that quantify the linear relationship between the predictions and the official rankings. The results indicate that both models performed well in predicting the top 100 players, with generally higher accuracy observed for players ranked close to the top. The Bradley-Terry model, which is based on a pairwise comparison of player performance, showed slightly better performance than the Elo ranking system in predicting the top 100 players for recent years. Further research could explore the use of additional factors, such as player injuries and weather conditions, to further enhance the accuracy of the models.

Keywords: ATP, Ranking system, Bradley-Terry model, Elo rating system, Kendall and Spearman's correlation coefficient

Table of Contents

List of Figures

List of Tables

1 Introduction

Tennis is a well renowned sport watched by many people across the world. Many large public tournaments are held such as Wimbledon, The American Open and The French Open, and the US Open where many male tennis players compete with each other to climb the ATP (Association of Tennis Professionals) rankings, with aspirations to be the top-ranking tennis player, as being in the top 100 players can be seen as a significant career milestone.

The current ATP ranking system for tennis players uses a point system and is frequently updated and altered based on players' match history in the past 52 weeks. The results of games between 2 players form pairwise comparisons, from which the rankings should be constructed. Formally, a paired comparison experiment collects independent judges' preferences towards any 2 objects (David 1963), but the idea can be extended to sporting disciplines. Tennis players have certain traits that physically and intangibly affect the results of their matches, and thus are suitable multi-dimensional subjects to apply techniques that deal with comparative judgement data on.

Extensive work has been made in this particular area of research and we will focus our attention on the **Bradley-Terry** (BT) model and the **Elo-Rating system**. We will first discuss how our raw data, collected from the tennis-data website, is pre-processed. After that, we will start by introducing the basic BT model and Elo system, outlining the calculations and assumptions, and then explain how the type of tournament, the round of the game, and match characteristics (court and surface type) are integrated into our simple models.

To validate the models, we evaluate these 2 models' fit by comparing the outputted rankings for years 2000, 2004, 2008, 2012, 2016, and 2019 with the published top 100 players at the end of the above 6 years. After considering the properties the ranking data have, we calculated Kendall and Spearman's correlation coefficients for both fitted and actual rankings of the top 100 and top 10 players for different 6 years. At the same time, we plot predicted ranks vs the published ones and inspect for prediction and error patterns.

We found from the results that in earlier years both models performed fairly similarly, however from 2016 onward, the Bradley Terry Model performed better in predicting the top 100 players. The Elo Model however, is simpler to use while the Bradley Terry model is more computational costly when handling large datasets.

2 Methods

2.1 Data Preprocessing

The original data set, consisting of 23 separate CSV files, contains the game and betting results for the men's ATP tour dating back to January 2000, including Grand Slams, Masters Series, Masters Cup and International Series competitions. We first discarded the columns related to betting odds, and then concatenated useful columns from each of the files to form an integrated table. After that, we applied the following steps to pre-process the data catering to the need for data exploration and modelling:

- Drop the tournament number and location columns, since they change according to the change of the name of the tournament and provides no additional information for our ranking system.
- Drop the columns containing the number of games won by the winner/loser in different sets, since we are only going to use the number of sets won by winner and loser, rather than the number of games won by each player, to rank the players.
- Drop the comment of the matches, which only shows if the game was completed, won through the retirement of loser, or via walkover. We can distinguish these 3 types of game simply from the number of sets won by each player.
- Change the data types to make sure each column has the correct type.
- Create a column from the entry ranking difference of the winner and loser, so that we can better analyse and incorporate the current abilities of the players in the models.
- The current ATP ranking system ranks players based on the matches played in the last 52 weeks, so the data set should be divided according to the time period each match belongs to. We can use many arbitrary 52 weeks from 2000 to 2023, but we choose to divide the whole time period into consecutive 52 weeks and form subsets of data based on that.

After data processing, every subset of data contains the name of the tournament, date of the match, name of the ATP series, type of surface and court, round of match, Winner's and loser's name, entry rankings of winner and loser, entry points of winner and loser, the number of sets won by winner/loser, the entry rank difference of winner and loser.

In order to compare the outputs of our models to the published ranking, we found additional data including the top 100 ATP players at the end of each week from 2000 to 2019. The only data manipulation we did for that data was changing the format of names of the players exactly the same as the formats in the match data set, e.g. Pete Sampras to Sampras P., to ensure that we can easily compare the fitted and published rankings of the same player.

2.2 Bradley-Terry Model

We first chose to use the Bradley-Terry model due to its ability to handle pairwise comparisons and its flexibility in accommodating different sources of data. The Bradley-Terry model is a popular method for modelling pairwise comparisons, such as tennis matches, where the outcome of a match is based on the relative strength of the opponents. The model assumes that the probability of player i beating player j in a match is proportional to the ratio of their underlying strengths. The model can be extended to handle ties and other complex scenarios, and can also incorporate additional information such as home advantage, surface type, or player-specific characteristics.

2.2.1 Standard Bradley-Terry

Based on the Bradley-Terry model (Turner and Firth 2012), two players in each game are denoted as i and j. In this case, every player plays every other player a different number of times, so players have different "strengths of schedule", meaning that some players play stronger opponents more frequently than other players. These players might have worse win-loss records, but in fact, be better than other players that won more games against weaker opponents.

Let β_i represent the 'strength' of player i and let the outcome of a game between players (i, j) be determined by $\beta_i - \beta_j$. The Bradley-Terry model treats this outcome as an independent Bernoulli random variable with distribution Bernoulli(p_{ij}), where the log-odds corresponding to the probability p_{ij} that team i beats team j is modelled as:

$$\log \frac{p_{ij}}{1 - p_{ij}} = \beta_i - \beta_j \tag{1}$$

Equivalently, solving for p_{ij} yields:

$$p_{ij} = \frac{e^{\beta_i - \beta_j}}{1 + e^{\beta_i - \beta_j}} = \frac{e^{\beta_i}}{e^{\beta_i} + e^{\beta_j}} \tag{2}$$

If we always order each pair (i,j) so we assume that player i is the player with a higher rank and j is the one with a lower rank, then we may incorporate an advantage coefficient in (1) by including an intercept term α:

$$log\frac{p_{ij}}{1 - p_{ij}} = \alpha + \beta_i - \beta_j \tag{3}$$

It is worth emphasizing that this model has the following property:

- The outcomes of each set of matches are independent Bernoulli random variables.
- Setting $\beta_i \equiv 0$ for a particular player i to solve the problem of over-parameterization. Then for every other player j, $\beta_j = \beta_j - 0$ represents the log-odds that player j beats player i.
- We assume the sample size is large enough such that the Maximum Likelihood Estimators of $\beta_i - \beta_j$ is asymptotically normal.

2.2.2 Statistical Inference of the Bradley-Terry Model

Let k be the number of players in one data set,our task is to:

- Estimate the advantage coefficient α and the team strengths $\beta_1, ..., \beta_k$.
- Test the null hypothesis of no advantage effect, $\alpha = 0$
- Obtain a confidence interval for $\beta_i - \beta_j$ for two particular players i and j.

Suppose we observe n total games (i_1, j_1),.....,(i_n, j_n) between these k players, where each (i,j) is a pair of distinct players in 1,...,k and the player who has advantage is player i. Let Y_1,...,Y_n 0,1 be such that Y_m if i_m beat j_m in the mth game and $Y_m = 0$ otherwise. The likelihood for the parameters $\theta = (\alpha, \beta_2, ..., \beta_k)$ is given by

$$\text{lik}\,(\alpha, \beta_2, \ldots, \beta_k) = \prod_{m=1}^{n} p_{i_m j_m}^{Y_m} \left(1 - p_{i_m j_m}\right)^{1-Y_m} = \prod_{m=1}^{n} \left(1 - p_{i_m j_m}\right) \left(\frac{p_{i_m j_m}}{1 - p_{i_m j_m}}\right)^{Y_m} \tag{4}$$

where p_{ij} is given as a function of α, β_i and β_j by the above equation and we set $\beta_1 \equiv 0$. Then deriving the log-likelihood and setting the partial derivative with respect to each parameter equal to 0:

$$0 = \frac{\partial l}{\partial \alpha} = \sum_{m=1}^{n} Y_m - \frac{e^{\alpha + \beta_{i_m} - \beta_{j_m}}}{1 + e^{\alpha + \beta_{i_m} - \beta_{j_m}}} \tag{5}$$

$$0 = \frac{\partial l}{\partial \beta_i} = \sum_{m:i_m = i} \left(Y_m - \frac{e^{\alpha + \beta_{i_m} - \beta_{j_m}}}{1 + e^{\alpha + \beta_{i_m} - \beta_{j_m}}}\right) + \sum_{m:j_m = i} \left(-Y_m + \frac{e^{\alpha + \beta_{i_m} - \beta_{j_m}}}{1 + e^{\alpha + \beta_{i_m} - \beta_{j_m}}}\right) \tag{6}$$

This yields a system of k equations in the k unknowns parameters, which may be solved numerically using the Newton-Raphson algorithm. The solution is the MLE $\hat{\theta} = (\hat{\alpha}, \hat{\beta}_2, \ldots, \hat{\beta}_k)$

To test the null hypothesis $H_0 : \alpha = 0$, we may use the generalized likelihood ratio test (GLRT): Under the sub-model where $\alpha = 0$, using the same steps to obtain the sub-model MLEs $\hat{\beta_{2,0}}, \ldots, \hat{\beta_{k,0}}$. The GLRT of $\alpha = 0$ is based on the test statistic

$$-2log\Lambda = -2log\frac{\text{lik}\,(0, \beta_{2,0}, \ldots, \beta_{k\text{fi}0})}{\text{lik}\,(\alpha, \beta_2, \ldots, \beta_k)} \tag{7}$$

and an approximate level-0.05 test rejects H_0 when $-2log\Lambda > \chi_1^2(0.05)$. We may obtain a confidence interval for $\beta_i - \beta_j$ by centering it around $\hat{\beta}_i - \hat{\beta}_j$, and estimating the standard error of $\hat{\beta}_i - \hat{\beta}_j$. Firstly considering the sampling distribution of MLE estimates $\hat{\theta} = (\hat{\alpha}, \hat{\beta}_2, \ldots, \hat{\beta}_k)$, when the number of the total games n is large, this is approximately $\boldsymbol{N}(\theta, \boldsymbol{I}_{\mathbf{Y}}(\theta)^{-1})$, where $\boldsymbol{I}_{\mathbf{Y}}(\theta) = -\mathbb{E}_\theta[\bigtriangledown^2 l(\theta)]$. It is easy to see that $\bigtriangledown^2 l(\theta)$ is a constant quantity that does not involve $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n$, so $\boldsymbol{I}_{\mathbf{Y}}(\theta) = -\bigtriangledown^2 l(\theta)$. Finally, since $\hat{\beta}_i - \hat{\beta}_j$ is a linear combination of the coordinates of $\hat{\theta}$, it is approximately normal when $\hat{\theta}$ is approximately multivariate normal. Then its variance is

$$Var[\hat{\beta}_i - \hat{\beta}_j] = Cov[\hat{\beta}_i - \hat{\beta}_j, \hat{\beta}_i - \hat{\beta}_j] \;\; \approx \boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{ii} + \boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{jj} - 2\boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{ij} \tag{8}$$

We may estimate the standard error of $\hat{\beta}_i - \hat{\beta}_j$ by the plug-in estimate

$$\hat{se}_{ij} = \sqrt{\boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{ii} + \boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{jj} - 2\boldsymbol{I}_{\mathbf{Y}}^{-1}(\theta)_{ij}} \tag{9}$$

2.2.3 Weighted comparisons

According to the above method, the data needs to be organized into pairs (player i, player j) and the corresponding win-loss frequency of player i vs. player j. The score for each game will be used to indicate the frequency of wins and losses. Since the winners and losers obtain different ATP points for different rounds of different events, we multiply the scores of each game by different coefficients according to the existing ATP points rules (Dexter 2023), so as to take the factor of "different types of tournaments" consider in our model.

In order to avoid overly complicating our model, the types of events are only divided into Grand Slam and non-Grand Slam when multiplying the weighting coefficient. After counting the number of non-Grand Slam events in recent years, which are ATP1000(9 events), ATP500(15)and ATP250(30+), the weighting coefficient of non-Grand Slam events uniformly adopt the points rules of ATP500 events.

Table 1: ATP ranking points table

Tournament Level	Winner	The Final	SF	QF	R16	R32	R64	R128
Grand Slam	2000	1200	720	360	180	90	45	10
ATP 500	500	300	180	90	45	20	-	-

[1] SF:Semifinals; QF:Quarterfinals

From the above table, it is obvious that for the vast majority of tournaments, Grand Slams' score are four times as many points as non-Grand Slams'. This is used to determine the weighted coefficient of each round of different types of tournaments. The following table gives examples of the coefficients under different situation:

Table 2: Weighted Coefficients for Different Tournaments

Tournament	Series	Round	Score	Weighted	After weighted
Australian Open	Grand Slam	R64	3:1	10	30:10
Australian Open	Grand Slam	Quarterfinals	3:1	20	60:20
Australian Open	Grand Slam	The Final	3:1	60	180:60
Dubai Open	non-Grand Slam	Semifinals	2:0	7.5	15:0
Dubai Open	non-Grand Slam	The Final	2:1	15	30:15

Although the coefficients do not follow the rule of ATP points strictly, it can take into account the influence of different tournaments on player rankings into the model.

2.2.4 Advantage Coefficient

In tennis matches, the player with a higher current ranking may have an advantage to his opponent. The amount of entry rank difference between two players could be considered to influence the degree of the advantage the higher-ranked player has. In addition to that, the advantage coefficient should also take the performance of players on different

match characteristics (court materials) in to account. Hence, we choose to use z_1 and z_2, both of which are between 0 and 1, to represent the current ranking advantage and the performance advantage due to different match characteristics respectively. The intercept term α in Formula (3) can be re-formulated as follows:

$$log\frac{p_{ij}}{1-p_{ij}} = \alpha z_1 z_2 + \beta_i - \beta_j \tag{10}$$

or equivalently

$$p_{ij} = \frac{e^{\alpha z_1 z_2 + \beta_i - \beta_j}}{1 + e^{\alpha z_1 z_2 + \beta_i - \beta_j}} \tag{11}$$

This increases the log-odds of the player with advantage winning in every game by a value of $\alpha z_1 z_2$.

Ranking Advantage According to the following figure, when the rank difference are less than 75, there is an approximate linear relationship between the rank difference and the winning probability of the player with a higher rank. However, this effect tends to saturate when the rank difference reaches 75 places. This saturation is noticeable in both Grand Slam and "Regular" (that is - non-Grand Slam) tournaments.

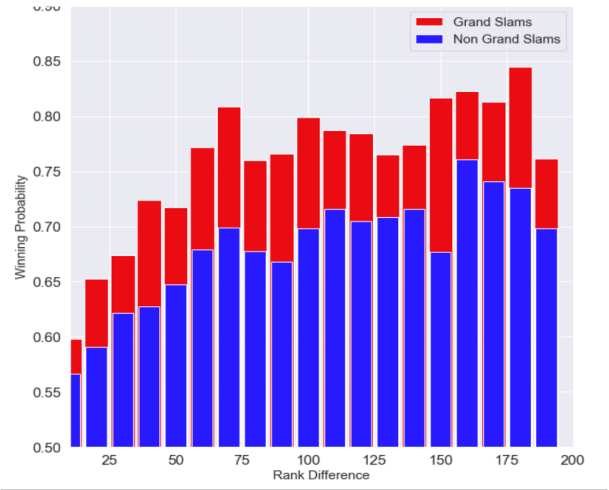

Figure 1: Winning Probability vs Rank Difference

Let $RD_{i,j}$ be the rank difference between two players i and j. Hence, the rank advantage z_1 caused by the rank difference between two players can be shown as:

$$z_1 = \begin{cases} \frac{RD_{i,j}}{75} & \text{if } 0 < RD_{i,j} \leq 75 \\ 1 & \text{if } RD_{i,j} > 75 \\ 0 & \text{otherwise} \end{cases}$$

Considerations about Match Characteristics The match characteristics being considered are court and surface types, and there are 6 possible combination of these 2 match charateristics, comprising indoor hard, indoor clay, indoor carpet, outdoor hard, outdoor

clay and outdoor grass. To integrate match characteristics into our model, we first calculate the winning probability of player i (higher-ranked player from 2.2.1) in each pair (i,j) under different match characteristics, using all the historical data before the beginning of the 52th week we rank the players on.

After getting the winning probability of player i while playing against player j on different match characteristic combinations for every possible pair of players, we define how z_2 is calculated. Let q denote the proportion of the combinations of court and surface types on which the player with a higher entry rank won against his opponent, with probability greater than or equal to 0.5. q equals to 1 means that player i, who has a ranking advantage, outperforms j on all types of courts and surfaces. Hence $z_2 = 1$ when $q = 1$, which means match characteristics advantage completely agrees with player i's ranking advantage. In the same way, $q < 0.5$ represents that player i's dominance in the ranking doesn't match his average performance on differenct court and surface types, so z_2 will be 0. In other cases, z_2 is equal to q, which means that the ranking advantage of player i is partially recognized as a manifestation of strong ability.

The mathematical expression of z_2 is as follows:

$$z_2 = \begin{cases} 1 & \text{if } q = 1 \\ 0 & \text{if } q < 0.5 \\ q & \text{otherwise} \end{cases}$$

The following table shows the player Almagro N.'s winning probability against Hanescu V. calculated from the whole data before 2020. The match characteristics that has not been played on is represented by '-'.

Table 3: Winning probability of i against j on various match characteristics

Player i	Player j	Match Characteristics	Winning Probability
Almagro N.	Hanescu V.	Indoor Hard	-
		Indoor Clay	0.666666667
		Indoor Carpet	-
		Outdoor Hard	1
		Outdoor Clay	0.722222222
		Outdoor Grass	-

In this example, there are only three possible combinations of surface and court with game records, and the winning probabilities of the player i (Almagro.N) on all these three types are all greater than 0.5, so $q = 3/3 = 1$, then $z_2 = 1$ in this case.

2.3 Elo Rating System

Another method that we can use to model the ratings of tennis players is to adopt an Elo system which would rank the players based on their current skill levels. It is already a widely used system which is used in many competitive two-player, zero-sum games, such as chess. The Elo system infers the relative skill levels of players based on their match history. Their rating is represented by a score, which is computed by previous matches and tournaments, reflecting the players' performances in past games. This score is then updated according to the outcome of their current games. The number of points gained or lost in their score would depend on the difference in rating between the players and the match's outcome. The player with a higher Elo rating is regarded as the stronger player.

For example, stronger players with a higher Elo rating would have a higher probability of winning than a weaker player with a lower Elo rating. Due to the consideration of expected performances of players, the Elo rating system would award a lot of points to a weaker player when winning against a stronger player, whereas if the stronger player wins which is expected, this results in lower rating changes to both players.

2.3.1 Elo Rating Assumptions

To carry out this the Elo rating model, assumptions are made. One assumption is that each player has a expected probability of winning a match based on the difference in skill level between two players which is estimated using a logistic function.

We also assume that the outcome of one match doesn't affect the outcome of any other match, so a player's performance in one match is unaffected by their previous matches or the matches of other players. This means that the player's rating reflects their overall skill level, regardless of any particular match outcomes.

Another assumption is that when a player wins a match, they are assumed to have a higher level of performance in comparison to their opponent but the Elo rating system is unable to look at their skill across the match, but only rather the outcome of the match being win or loss. Therefore, their performance level is only inferred.

2.3.2 Elo's Expected Probability

In a tennis match between two players, Player A and B, we have that $E_A + E_B = 1$ where E_A denotes the expected number of points given to Player A, and the points awarded are:

$$\begin{cases} 1, & \text{if Player A beats Player B} \\ \frac{1}{2} & \text{if Player A and Player B tie} \\ 0 & \text{Player A loses to Player B} \end{cases}$$

However in a game of tennis, the game cannot end in a draw as the match is played out until a winner is declared therefore we would disregard the points for a draw. Defining the

variable X_A as the number of points which is awarded to Player A when playing against Player B, E_A can be written as the expectation of X_A:

$$E[X_A] = E_A = 1P(A > B) + 0P(A < B) = P_A \tag{12}$$

Here we can see that the expected number of points awarded to player A is P_A which represents the probability of Player A winning against Player B.

The function used to determine the expected score would depend on the difference between the current Elo rating or underlying skill level between Player A and B which would be denoted as R_A and R_B. To derive the Elo system we would use the value 400 in the function as a scaling factor that determines the function's sensitivity to differences in player ratings. Here we would assume that a difference of 400 in rating points between Player A and B would mean that Player A would be 10 times more likely to win the game and so Player A's expected score is 10 times higher than Player B. We can then write this as:

$$\begin{aligned} E_A = 10^{(R_A - R_B/400)} E_B \Rightarrow E_A &= 10^{(R_A - R_B/400)}(1 - E_A) \\ \Rightarrow E_A &= 10^{(R_A - R_B/400)} - 10^{(R_A - R_B/400)} E_A \\ \Rightarrow E_A = P_A &= \frac{10^{(R_A - R_B/400)}}{1 + 10^{(R_A - R_B/400)}} \end{aligned} \tag{13}$$

We can then simplify this to obtain the function below which calculates the expected probability of Player A winning a game using both player's current Elo rating:

$$P_A = \frac{1}{1 + 10^{(R_B - R_A/400)}} \tag{14}$$

Where P_A represents the probability of Player A winning the game and R_A and R_B are the current Elo ratings of player A and B, respectively. This formula can be used for player B to calculate P_B, where in this case R_A and R_B are swapped around. This formula will provide a value between 0 and 1, where the larger the difference in their rankings, the larger the expected probability of winning.

This is formula is a standard logistic function of base 10 as shown by figure 2 below, which has the property of having an S-shaped curve. In the context of an Elo system, this property is important to highlight that the steep slope near the center of the curve means that small differences in skill between two players could have a large impact on the winning probability, while the gradual flattening of the curve at the top and bottom means that large differences in skill level between two players would have less of an impact on the winning probability.

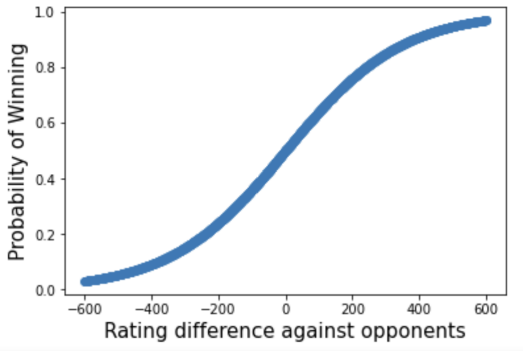

Figure 2: Relationship between the ranking difference and Probability of Winning

2.3.3 The Rating Update

The outcome of a match would influence the ratings of the players depending on whether the match outcome is better or worse than their expected probability of winning. Hence, to calculate a player's new Elo rating after a match, the formula below can be used to update the rankings using the scaled difference in expected and actual performance of the match:

$$R_A^{'} = R_A + K(S_A - P_A) \tag{15}$$

Where $R_A^{'}$ is the updated rating of player A, R_A is the initial Elo rating of Player A before the game, K is the K-factor and S_A is the actual result of the match, being either 0 or 1. This signifies a loss or a win, respectively. P_A denotes the expected winning probability which is calculated by the Elo expected probability function above (14).

Using this function will calculate the updated ranking of any specific player after a game. This can be used for both the winner and loser of the match to alter their ranking. The K-factor is a scalar, which determines the maximum amount that the player's Elo rating can change by, where a larger K - factor has a larger effect in altering the player's ranking and a smaller K would result in insignificant changes.This factor can be varied based on the significance of the match or tournament, performance on different surface types or the number of games played which we will explore below when adding match characteristics.

Therefore, the Elo rating system would use a player's previous performances to predict the likelihood of winning future games and alter their rank from results of their following games. This means that over time, players with higher ratings would be expected to win more often, while lower ranked players would win less often, which leads to a self-correcting system that would converge to a stable set of rankings over time.

2.3.4 Match Characteristics in the Elo Model

Upon implementing a simple Elo rating model we chose to set a static K - Factor of 32 since this value was frequently used by Arpad Elo, who found that based on empirical observations, a K-factor of 32 produced the most accurate results in reflecting the player

strength and predicting future performance. The choice of 32 (S. Kovalchik 2020)has also later been validated via different studies in various sports and generally a K-Factor between 24-40 produced stable ratings over time.

As seen in the Bradley Terry model, some players have an advantage when playing on a specific type of court surface. Different match characteristics like this can also be taken into account for the Elo rating system by further changing the K factor or adding a specific weighting factor. For example, in the FIFA Football world Elo ranking system, the K-factor reflects the importance of a match by assigning a smaller value of K to friendly matches and a larger number for more important games such as the FIFA World cup final.

Similarly this could be implemented in our Elo rating system in order to account for the difference in players abilities when playing on different types of court surfaces, and importance of the tournaments so that the Grand Slams would be weighted higher than the non-Grand Slam games as we stated previously in the Bradley Terry Model. We would also assign a different factor for winning specific rounds in the tournament so that a higher K-factor is given to a finalist than the winner of a match in the 1st round.

We could change the ranking update function from the simple model by multiplying the K-factor by a weighted scale factor that represents the importance of Grand Slam which consists of games played in the Australian Open, the US Open, the French open and Wimbledon, over non - Grand Slam games which is every other tournament. The update rank function would now be:

$$R_A^{'} = R_A + [K \cdot WS_T \cdot (S_A - P_A)] \tag{16}$$

Where WS_T is the weighted scale factor based on the tournament and round.

We once again chose to split up the tournaments into these two groups to avoid over-complicated the model, but also ensure the factors are considered in the same way for both the Bradley Terry and Elo models. From the official ATP ranking points in Table 1, we saw that for majority of tournaments, Grand slams' scores were four times higher than that of non Grand Slams'. This is considered when determining our own weighting scale for each round of tournaments which is as follows:

Table 4: Weighted Scale Factor for Different Tournaments

Round	Grand Slam	Non - Grand Slam
R128	0.2	N/A
R64	0.6	N/A
R32	0.8	0.2
R16	1.6	0.4
Quarterfinals	2.4	0.6
Semifinals	3.2	0.8
The Final	4.0	1.0

Since, there are a lower number of players competing in non-Grand Slam games, there

isn't a weighting factor associated with R128 and R64. These weighting factors will allow us to take into account both the tournament's importance and the round reached by a player.

A similar approach can be taken to allow our model to consider the differences in players performances on different types of courts as the surfaces differ. Players may perform better on a specific court surface in comparison to another as each surface has different playing characteristics which would suit their individual style of play. The four types of surface include: Clay, Grass, Hard, Carpet. To integrate surface types, we would have to change the update function once again to include a surface-specific weighting factor, as shown below:

$$R_A^{'} = R_A + [K \cdot WS_T \cdot WS_S \cdot (S_A - P_A)] \tag{17}$$

This would be our final model, where WS_S is the weighted scale factor based on the surface Player A plays the match on.

However the surface specific weighted factor is determined very differently as this is specific for every player. So here WS_S would be reflected by the winning player's historical performance on that surface. We would first need to calculate the percentage of sets won by each player on different surfaces for Player A, by using historical data before the beginning of the 52th week we rank the players on and compare it their percentage of sets won on all surfaces to get a ratio between them which gives us the surface-specific weights. A weighting factor greater than 1 indicates that the player performs better on that surface, while a weighting factor less than 1 indicates that the players perform worse on that surface. The following table below shows the win-loss percent and weights for Thiem D. calculated from the whole data set before 2019:

Table 5: Win-Loss Percentage for an example Player A on different surfaces

Player A	Surface Type	Win-Loss Percentage	Weight
Thiem D.	Hard	0.758628	0.8348
	Clay	0.951557	1.1470
	Carpet	0.857143	0.9532
	Grass	0.846154	0.9411

When coding for this model on R for ranking the players at the end of 2019, we used all data available before 2019, to calculate the win - loss percentages and thus the surface specific weights for each player. The match data from 2019, specified the name of the winner and loser, the tournament played in, whether the game was a grand slam or not, the round of the tournament and the surface it was played on into order to associate the correct weighting factors corresponding to the tournament played in and the players performance on that surface played in that match.

2.4 Model Validation

To compare the published data with the ranking data predicted by the Bradley-Terry model and Elo Rating System, we have to consider the following properties in the first place to choose suitable validation methods:

- Rankings are ordinal integers and possess the properties that intervals between adjacent units are equal. Equal intervals mean that there are equal amounts of the variable being measured between adjacent units on the scale.
- The predicted data is monotonically related to the published data, which suggests that some correlation coefficients might be informative to show the similarities between these 2 data sets.
- The data is not normally distributed, violating the assumption of Pearson's correlation coefficient.
- Performance measurements that incorporates distance calculation, e.g. mean square errors, are not perfect choices for showing the rank differences between the predictions and the real ranks, because ranking the top 1 player to the 100 place considerably worse than ranking a 100-placed player to 200.

Based on the above properties, we ruled out Pearson Correlation and chose Kendall Correlation and Spearman Correlation, both of which are non-parametric measurements. Kendall correlation is preferred when there are small samples or some outliers while Spearman Correlation is more suitable for dealing with large-scale data. Since our fitted data size is moderate, we use both of validation methods.

2.4.1 Kendall Rank Correlation Coefficient

Kendall rank correlation (τ) is a statistic used to measure the ordinal association between two measured quantities. A τ test is a non-parametric hypothesis test for statistical dependence based on the τ coefficient.

We have got two samples, which are the predicted data set and the published data set, where each sample size is n. We know that the total number of pairings with two samples is n(n-1)/2. The following formula is used to calculate the value of Kendall rank correlation coefficient:

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n-1)} \tag{18}$$

n_c = number of concordant; n_d = number of discordant

From the above formula, it is obvious that the coefficient is between -1 to 1. And it is said to be strong relationship if the coefficient is greater than 0.6.

2.4.2 Spearman's Rank Correlation Coefficient

Another measure for correlation of ranks developed by Spearman (1961) is a well-known example to compare the model-predicted ranking to the players' actual ranking. It is given by

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \tag{19}$$

d_i = the difference between the ranks of corresponding variables

In addition, a significant test for the result can be constructed by the statistic

$$t = \frac{\rho\sqrt{n-2}}{\sqrt{1-\rho^2}} \tag{20}$$

which follows a Student t distribution with $n - 2$ degree of freedom.

As Spearman rank correlation is a non-parametric test, it is unaffected by how the population is distributed. This makes it viable for smaller sample sizes. Correlation coefficients between 0.30 and 0.49 represent a medium association, and coefficients of 0.50 and above represent a large association or relationship.

3 Results

In the results section, to evaluate the overall performance of our models, we representatively select out 6 52-weeks' data (DataHub.io, 2019) and run our ranking systems on each of these 6 subsets of data. With the consideration of the fact that entering top 100 in ATP rankings can be regarded as a career mile stone for tennis players, we found the top 100 players at the end of every week between 2000 and 2019 to compare with the fitted rankings from our models. Since the published ranking ends at 2019, we equally spaced these 20 years and select 2000, 2004, 2008, 2012, 2016, and 2019 to run our models on.

We are going to firstly show the outputted results (e.g. estimates, standard errors) from our models, and then in the validation section, published ranking will be compared with the fitted ones using visualization and correlation coefficients.

3.1 Bradley-Terry Model Outputs

In this section we apply the Bradley-Terry Model to real data and output the predicted rank of 6 years respectively. Computations, including numerically finding the MLEs, were done in **R**, using the **BradleyTerry2** package (Turner and Firth 2012) We firstly use this package to computes estimates for the log-ability scores $\beta_1, ..., \beta_k$ of the simple Bradley-Terry model.

Then advantage coefficient α is considered into the simple model and computed year by year, bringing in the ranking advantage z_1 and the performance advantage in different match characteristics z_2 we get the advantage coefficient for each 6 years.

Table 6: Advantage coefficients α of BT model in 6 years

Year	Advantage coefficients α	p-value
2000	-0.39022	1.03×10^{-7}
2004	-0.36120	$< 2 \times 10^{-16}$
2008	-0.57026	$< 2 \times 10^{-16}$
2012	-0.42260	$< 2 \times 10^{-16}$
2016	-0.35636	2.15×10^{-11}
2019	-0.32470	1.02×10^{-12}

In football or basketball games, if the home team is given an advantage α, then α is generally positive since the home team has home advantage which can positively effect the performance of itself. A positive coefficient removes the winning probability brought by the home field advantage, making the team's ability prediction more accurate.

However, in our case, a negative value of α indicates if the winning probability p_{ij} in formula (10) is fixed, i.e., the total outcome of the games they have played together are the same, the ability difference between player i and j will be enlarged. This further affirms that when the weak player has a certain ability, the stronger player has the ability corresponding to his ranking advantage and performance dominance, and this can also

improve the accuracy of $\beta_1, ..., \beta_k$. And the corresponding p-value shows all the outputs are significant.

As what have been mentioned in the method section, these ability scores β are invariant to additions/subtractions, and only the difference between the scores are important. The function **BTm** in the package randomly chooses one player as the reference whose estimates will be 0, other players' abilities are relative values of this player's ability.

As an example of the model outputs, the following table contains the top 10 player at the end of 2016 computed by the Bradley-Terry model after considering the advantage coefficient.

Table 7: Top 10 player rankings according to Bradley-Terry model from ATP data

	player	ability(β)	standard error
1	Djokovic N.	1.6703	0.08340952
2	Murray A.	0.9651	0.07914988
3	Wawrinka S.	0.8115	0.08745546
4	Federer R.	0.6271	0.09237075
5	Nadal R.	0.3727	0.09791413
6	Raonic M.	0.3386	0.08529792
7	Monfils G.	0.2586	0.09757145
8	Nishikori K.	0.2560	0.08774491
9	Bourgue M.	0.2033	0.30341333
10	Berdych T.	0.0000	0.00000000

The above table gives the estimated ability for each player when the player enjoys no advantage and displays the top-10-rated players from our data. This ranking follows the natural convention that the higher a player's β, the better we assume the player to be.

To visualize the output estimates with standard error from the BT model, we plot the figure of estimated relative abilities for 50 players selected randomly.

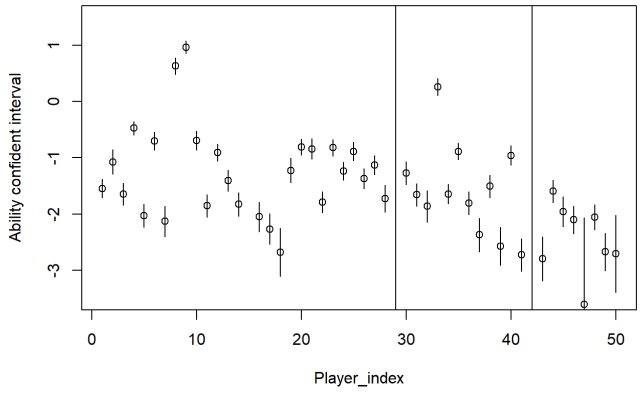

Figure 3: Estimated relative abilities of 50 tennis players

The 'comparison intervals' shown are based on 'quasi-standard errors' and can be interpreted as if they refer to independent estimates of player ability. This has the advantage that comparisons can be made easily for any pair of players (i.e. not only with 'reference' players). It is obvious to observe that the standard error of few players are quite large, and the reason for this result is that some players only participated few tournaments so the battle data is not sufficient.

3.2 Bradley-Terry Model Validation

In this section, we validate our model via comparing the ranks from outputted ability scores with the published rankings for year 2000, 2004, 2008, 2012, 2016, and 2019. We first found the fitted rankings of the actually top 100 players, and then calculate the Kendall and Spearman's correlation coefficients between these 2 vectors of ranking. After that, we plotted the predicted ranks vs the published ranks and realized that in most years, the Bradley-Terry model performs better while finding the top ranked players, so we then calculated those 2 correlation coefficients for only the top 10 players.

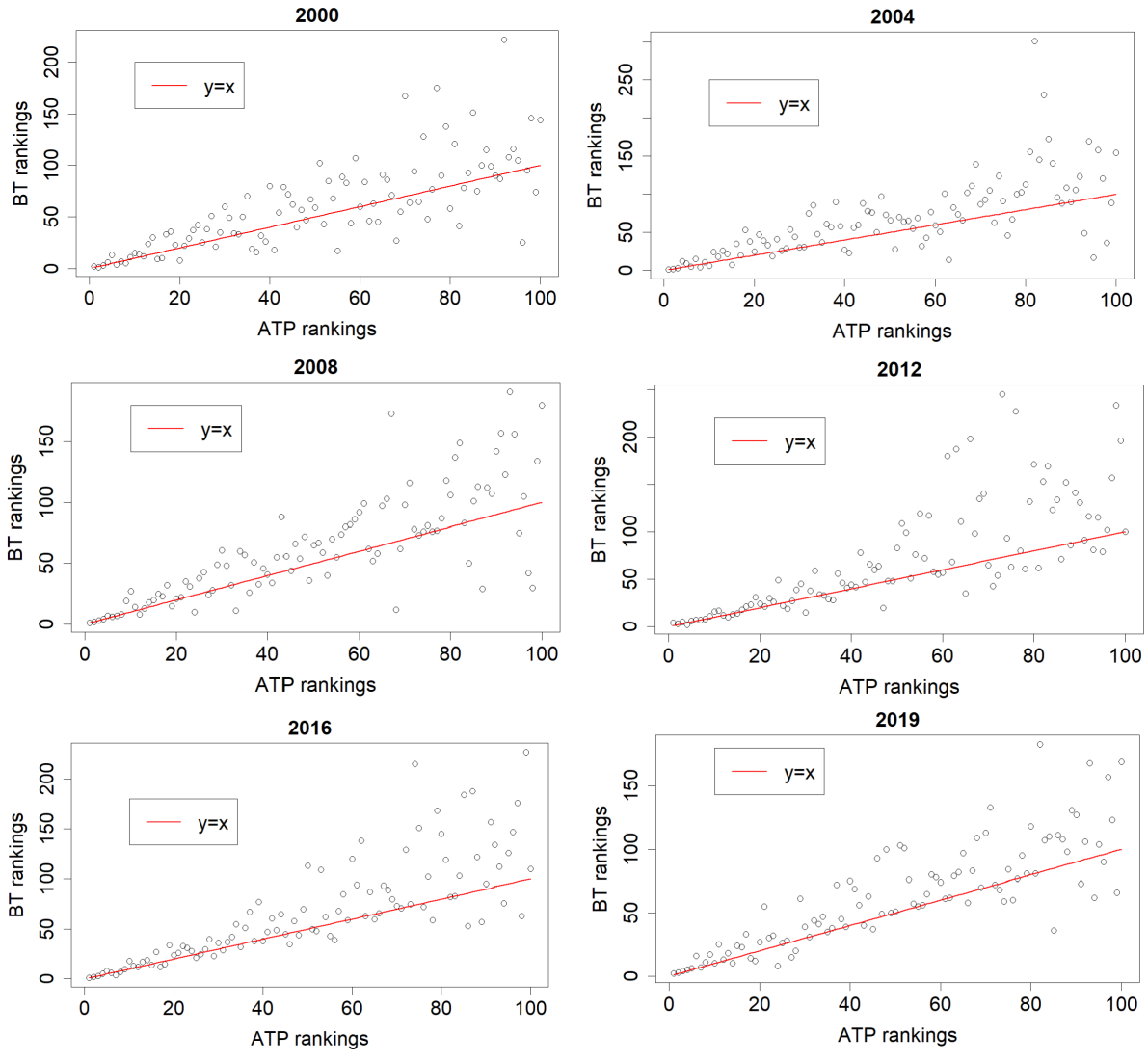

Figure 4: Fitted ranking vs published ranking for 2000, 2004, 2008, 2012, 2016, and 2019

Table 8: Kendall and Spearmans' correlation coefficients for BT model

Year	τ_{100}	τ_{10}	ρ_{100}	ρ_{10}
2000	0.6362	0.6444	0.7642	0.7324
2004	0.6209	0.4222	0.7122	0.4598
2008	0.6853	0.9439	0.7937	0.8586
2012	0.6954	0.8090	0.7647	0.8681
2016	0.7332	0.7778	0.8135	0.8344
2019	0.7048	0.7778	0.8379	0.7712

From the above plots and correlation coefficients, the following bried comments can be made regarding the validation of our model:

- All of τ_{100}s are larger than 0.6 and all of the ρ_{100}s are larger than 0.7, indicating there are relatively strong linear relationships between the published top 100 rankings and the fitted ones for all the 6 years. We didn't give any p-values for the associated tests, since they are all very small, confirming the significant of these monotonic linear relationships. This result shows the level of accuracy of our ranking system is satisfactory and hence validates our model to a large extent.

- Looking at τ_{100}s and ρ_{100}s, we can find that they are higher in the 2016 and 2019 than in the previous 4 years. This might because the weights we gave for different rounds and tournaments are based on the scores current ATP ranking system allocates to different rounds and tournaments, and how the scores are decided are actually changing from year to year.

- Focusing on τ_{10}s and ρ_{10}s, we can found that expect in year 2004 and 2019, they are all higher than thier top 100 counterparts, indicating Bradley-Terry model generally performs better in terms of ranking top tier players. The main reason for this result is because only via not being eliminated and playing more games in one specific tournament, can one player obtain more points in the ATP ranking system, which means the players with high ranks have more game records than other players and the estimates for their abilities will subsequently become more accurate.

- Only in 2004, both of Kendall and Spearsman's coefficients drop from a relative satisfactory level to below 0.5 when they are calculated for published top 10 players instead of top 100 players. Closer inspection into the data reveals that many pairs of players' rankings are interchanged in the fitted rankings resulting in a low rank correlation coefficient.

- From Figure 4, we can further confirm some of the conclusions we made above. The points generally fall in the neighborhood of the diagonal line $y = x$ and the distance between the point and line $y = x$ gradually increases when the real ATP ranking increases, demonstrating the level of linear relationship between the fitted rankings and published ones is high in general and becomes even higher for top-ranked players. The reason why the players are more often been lower ranked rather than ranked to a higher position is simply because we are comparing top 100 players to their fitted rankings that might take value higher than 100, but not lower than 0.

3.3 Elo Rating System Outputs

In this section, we implement the Elo System by using both the Expected Probability function and the Ranking Update function by fitting the real tennis data-set to derive the rankings. We start of with the simple model which uses both equations (14) and (15) from earlier. As an example of the model outputs, the following table contains the top 10 player at the end of 2019 computed by the simple Elo rating system model:

Table 9: Top 10 player rankings according to the Elo rating system from ATP data

	player	Elo rating
1	Nadal R.	1989.56
2	Djokovic N.	1954.00
3	Federer R.	1935.18
4	Medvedev D.	1925.06
5	Thiem D.	1880.66
6	Tsitipas S.	1813.18
7	Berrettini M.	1799.28
8	Shapovalov D.	1792.71
9	Rublev.A	1782.63
10	Zverev A.	1769.39

This shows the top ten players at the end of 2019 without taking into consideration the importance of the tournament, the round which a player reaches in the tournament, and the surface type while only using a K-factor of 32.

In our final model, when taking into consideration the specific round reached, the importance of the tournament, and the different surface types which uses both equations (14) and (17), the results differ from that of the simple model as shown below by the models output of the top ten players at the end of 2019:

Table 10: Top 10 player rankings according to the Elo rating system from ATP data

	player	Elo rating
1	Nadal R.	1727.12
2	Medvedev D.	1718.70
3	Djokovic N.	1709.37
4	Thiem D.	1703.28
5	Federer R.	1687.62
6	Tsitipas S.	1656.78
7	Berrettini M.	1651.438
8	Bautista A.	1636.08
9	Balazs A.	1632.54
10	Nishikori K.	1629.43

Some differences can be seen between the two rankings. We can see that Medvedev D.

is ranked higher when taking in to account the tournament and surface. When scanning through the match data in 2019, we can see that he had reached very far in many of the tournaments, even in the grand slam which would have associated with a higher weight, and so resulted in a higher Elo rating. Nadal R. remained the best player in both models, having a higher win-loss percentage on all surface in comparison to other players, and had gotten very far in many of the tournaments also winning 2 grand slams, the French Open and the US Open.

3.4 Elo Rating System Validation

In this section we validate the Elo rating system model by also comparing the published rankings against our rankings across the years: 2000, 2004, 2008, 2012, 2016 and 2019. However for the 2019 data set, it is incomplete so there are missing data values. This will be taken into account when commenting on the results. We will be using the simple Elo model and also the complex Elo model to compare. From this, we can again calculate the Kendall and Spearman's rank correlation coefficients to validate our models. Two coefficients are found with each validation technique, one for all the top 100 players and one for the top 10 players.

3.4.1 The Simple Elo System Validation Results

Here are the results of the validation techniques used for our simple Elo rating system, where we only use the K-factor which is set at 32.

Table 11: Kendall and Spearmans' correlation coefficients for the simple Elo system

Year	τ_{100}	τ_{10}	ρ_{100}	ρ_{10}
2000	0.5825	0.2889	0.7795	0.3312
2004	0.6513	0.6889	0.8460	0.7470
2008	0.6858	0.7502	0.8637	0.8434
2012	0.6241	0.8090	0.7938	0.8428
2016	0.6516	0.6889	0.8373	0.8061
2019	0.5976	0.8090	0.7817	0.7087

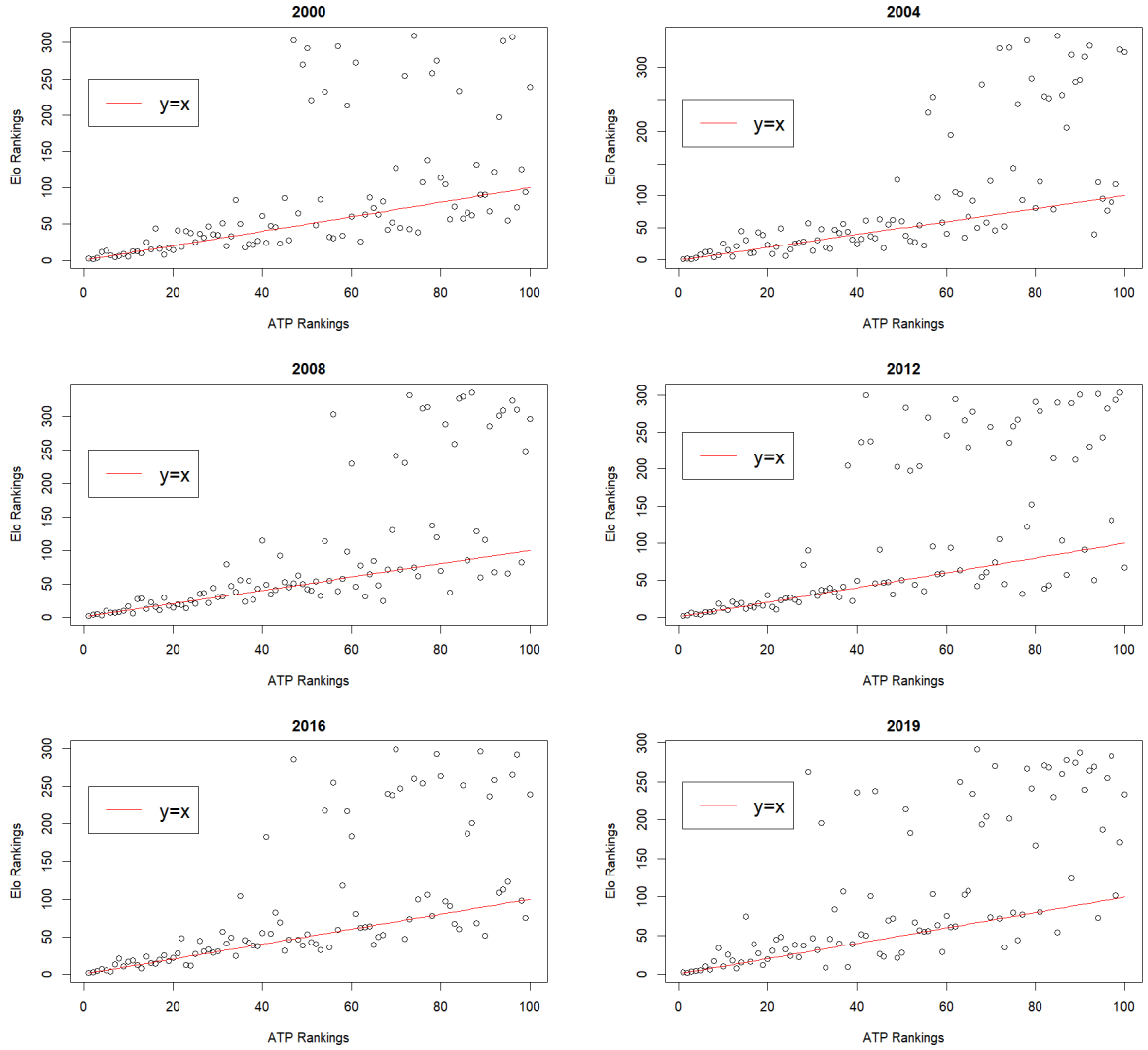

Figure 5: Fitted simple Elo ranking vs published ranking for 2000, 2004, 2008, 2012, 2016, and 2019

From the above information, we can make some comments regarding the simple Elo model:

- Most τ_{100} are larger than 0.6 and all ρ_{100} are larger than 0.7. This presents a strong linear association between the fitted simple Elo ranking system and the published ATP rankings. This validates our simple Elo ranking system as the level of accuracy is relatively high. As with the Bradley-Terry model, the p-values were omitted in these results as they were all significantly small, which represents a strong monotonic linear relationship.

- When considering the top 10 players across the years, τ_{10} would hold a larger value than τ_{100} after the year 2000, indicating that the simple Elo model performs better for higher ranking players. This can be explained by the fact that higher ranking players often reach further in tournaments and so participate in more matches. Consequently, there will be more game records for these players and so the model will be able to calculate their performance through the Elo ranking system more accurately.

- The year 2000 has the lowest coefficient correlation values for the top 10 players across the years. Despite this, looking at figure 5 for the year 2000, we can see that the a lot of the players still reside on the $y = x$ line. However, upon further inspection, we see that the majority of the top 10 remains the same as the official ATP rankings, but there are a lot of interchanging ranking between players rankings.

3.4.2 Complex Elo Rating System Validation Results

Our complex Elo rating system takes into account some match characteristics. These include: how far a player reaches in a tournament. These can vary from the finals, semi-finals and even the first rounds. How important the tournament the player is participating in is. For example, the Grand Slam is considered the most prestigious tournaments for individual tennis players, hence winning these matches will result in a larger increase in the Elo ranking. The different surface types. This is heavily considered as some players may perform at a higher level on specific surfaces and so will have an advantage during their games against their opponents.

Only the 2019 ATP official rankings were used to validate the complex Elo model. This is because, it is the most recent set of data, therefore there is more historical data for participants playing under different surfaces.

Using these characteristics, and also the scaled K-factor that was proposed earlier for different tournaments we can validate our complex model to see not only how well it compares with published ATP rankings, but also against the simple Elo model.

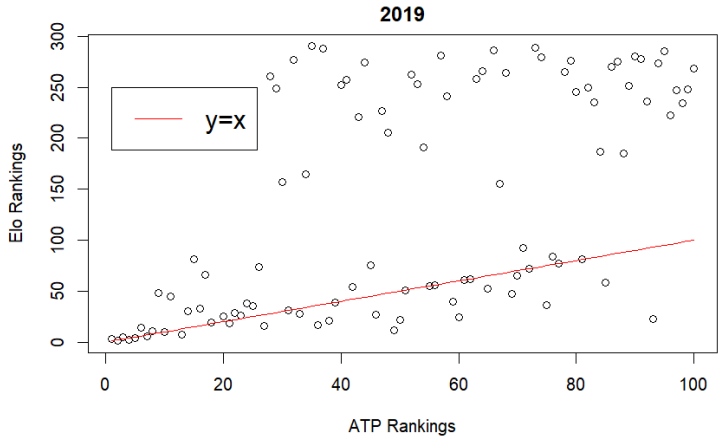

Figure 6: Fitted complex Elo ranking vs published ranking for 2019

Table 12: Kendall and Spearmans' correlation coefficients for the complex Elo system

Year	τ_{100}	τ_{10}	ρ_{100}	ρ_{10}
2019	0.4301	0.6000	0.5929	0.5876

From these results, we can see that from the plot there is a lot of distance from the points and the line of $y = x$, especially towards the lower ranked players. In addition,

the Kendall and Spearman's coefficient values are quite low in comparison to the values obtained in our simple Elo model for both the top 100 and top 10 players.These lower values can be explained by the fact that our model takes into account of multiple factors, whereas in the official ATP rankings, they do not. This implementation will cause players rankings to alter and shift around as their performance is now ranked by multiple factors that were not measured before.

Furthermore, higher ranked players have less variation in comparison to lower ranked players. This can be from the fact higher ranked players outperform lower ranked players more often and so progress further in tournaments including more important tournaments. And also these players have been trained to perform relatively well across all surface types, hence when playing against their opponent, they are able to sustain a high level of performance despite the flooring.

3.5 Comparison Of All Validation Results

We then compare the performances of all 3 models in 2019, and the following table gives all the correlation coefficients for 2019's rankings.

Table 13: Kendall and Spearmans' correlation coefficients for all the models in 2019

Model	Bradley Terry	Simple Elo	Complex Elo
τ_{100}	0.7048	0.5976	0.4301
τ_{10}	0.7778	0.8090	0.6000
ρ_{100}	0.8379	0.7817	0.5929
ρ_{10}	0.7712	0.7087	0.5876

It can be noticed that compared to the Bradley-Terry model, both the simple and complex Elo rating system outputs less correlated rankings with the published ranks, except τ_{10} for simple Elo. If we read back and compare the correlation coefficients between the simple Elo and BT models in all 6 years, the BT model only significantly outperformed the Elo system in 2016 and 2019, which are the most recent 2 years. In addition, since the Bradley-Terry model is more complex and requires more computational power, we thus cannot simply conclude that the Bradley-Terry model is the best choice.

4 Conclusion

We used both the Bradley Terry and Elo Models to devise the rankings of tennis players and used two validation techniques to compare the published rankings with the fitted rankings. We first started with basic models and then incorporate more influential factors into them. We measured the correlation of ranks from both Kendall and Spearmans' correlation coefficients. In general, both of the models give more accurate results while predicting the top-ranked players but still perform well on the top 100 players. The Bradley-Terry model performs the best while fitting 2016 and 2019 match data, but it isn't the best choice since the level of simplicity is sacrificed compared to the Elo system.

4.1 Limitations

- Weighted coefficient for the non-grand slam tournaments only depends on the points rule of ATP500.
- Rankings calculated from the Bradley-Terry Model is purely based on the point estimations of players' abilities, but the actual confident intervals of consecutively ranked players' ability scores are overlapping in most circumstances. This means the differences between some of the players' abilities are actually not statistically significant, but they are ranked differently in our system.
- The Elo system does not take into account players injuries and absences. Implementing this factor would mean altering player's Elo rank due to their specific circumstances. For example, if a player is injured or absent for a large duration of time, we would assume that their performance level would be lower, due to their physical limitations, by not participating in tournaments.
- Elo ignores the participant's playing style. Despite the fact that some players may perform better under certain match conditions, their performances against other specific players is disregarded. An example of this can be if Player A beats Player B and Player B beats Player C, then A would have a higher chance to beat C under the Elo model. However, in tennis, this is not always the case and Player C could in turn consistently beat Player A due to difference in playing style.
- The surface specific weight added using a player's historical win-loss ratio on a surface may not account for changes in a player's ability on that surface over time, so if a player improves their performance on clay over the course of the season, this will not be captured by their historical data on that surface.

4.2 Further Considerations

- The ATP ranking system has changed its rule of allocating points in the past 20 years.

- For the Bradley-Terry model, we could also approach it in a Bayesian manner rather than the frequentist way (Caron and Doucet 2012). By giving a prior assumption on the distribution of players' ability score, we can observe how the match results change the distribution.
- For the Elo rating system we can take use the number of games played throughout their careers up until the current tournament of ranking to consider the experience of a player and how that would affect the rankings in an Elo Model. This can be done by adapting a dynamic formula for the K-factor by Kovalchik (S. A. Kovalchik 2016).
- For the Elo system, instead of taking into consideration the winning percentage of a player for different surfaces and constructing an overall single ranking system, we could have split the data to rank players for each surface type to have separate rankings for each surface.
- Using other methods such as cross-validation, which compares each model's performance when introduced to unseen data, to validate our models.

References

Caron, Francois and Arnaud Doucet (2012). 'Efficient Bayesian inference for generalized Bradley–Terry models'. In: *Journal of Computational and Graphical Statistics* 21.1, pp. 174–196.

David, Herbert Aron (1963). *The method of paired comparisons*. Vol. 12. London.

Dexter, Robert (Jan. 2023). *Tennis ranking system: How does it work for WTA amp; ATP?* URL: https://supertennisracquet.com/tennis-ranking-system/.

Kovalchik, Stephanie (2020). 'Extension of the Elo rating system to margin of victory'. In: *International Journal of Forecasting* 36.4, pp. 1329–1341.

Kovalchik, Stephanie Ann (2016). 'Searching for the GOAT of tennis win prediction'. In: *Journal of Quantitative Analysis in Sports* 12.3, pp. 127–138.

Turner, Heather and David Firth (2012). 'Bradley-Terry Models in R: The BradleyTerry2 Package'. In: *Journal of Statistical Software* 48.9, pp. 1–21. DOI: 10.18637/jss.v048.i09. URL: https://www.jstatsoft.org/index.php/jss/article/view/v048i09.

Appendix

Data Preprocessing

⋮

```
#calcalculate weight
import datetime as dt
#fix data types
data['Date'] = pd.to_datetime (data ['Date'])
data.WRank = pd.to_numeric(data.WRank, errors = 'coerce')
data.LRank = pd.to_numeric(data.LRank, errors = 'coerce')

df['w'] = np.repeat(1,len(df))
def calculate_weight(tournament_name,DataFrame):
    data = DataFrame[DataFrame.Tournament == tournament_name]
    rounds = np.unique(data.Round)
    if list(data.Series.unique())==['Grand Slam']:
        for i in range(1,len(rounds)+1):
            DataFrame.w[(DataFrame.Tournament == tournament_name)&(
    DataFrame.Round == rounds[-i])] = 60/i
    else:
        for i in range(1,len(rounds)+1):
            DataFrame.w[(DataFrame.Tournament == tournament_name)&(
    DataFrame.Round == rounds[-i])] = 15/i
    return DataFrame

for t in np.unique(df.Tournament):
    df = calculate_weight(tournament_name=t,DataFrame=df)

#winning rate (different match characteristics)
df['match_char'] = df.Court + '_' + df.Surface
df.match_char.unique()
df['win_rate']=df['Wsets']/(df['Wsets']+df['Lsets'])

end_time = dt.datetime (2020, 1, 1, 0, 0, 0)
df_2019 = df[df['Date'] <end_time]
w1 = pd.DataFrame(df_2019.groupby(by=['match_char','Winner','Loser']).
    win_rate.mean()).reset_index()
pd.DataFrame(w1.groupby(by=['players','match_char']).win_rate.mean())
```

Bradley-Terry Model

⋮

```
### basic model
df <- read.csv('Data_1.csv')
library(dplyr)
library(BradleyTerry2)

basic_df <- df %>% select(Winner,Loser,Wsets,Lsets)

basic_df$W <- factor(basic_df$Winner, levels=unique(c(basic_df$Winner,
    basic_df$Loser)))
basic_df$L <- factor(basic_df$Loser, levels=unique(c(basic_df$Winner,
    basic_df$Loser)))

```

```
basic_model <- BTm(cbind(Wsets,Lsets),W,L,id='player_',data=basic_df)

Abilities <- data.frame(BTabilities(basic_model))
Abilities[order(-Abilities$ability), ]

### with weights
basic_df <- df %>% select(Winner,Loser,Wsets,Lsets,w)

basic_df$w <- as.numeric(basic_df$w)

basic_df$W <- factor(basic_df$Winner, levels=unique(c(basic_df$Winner,
    basic_df$Loser)))
basic_df$L <- factor(basic_df$Loser, levels=unique(c(basic_df$Winner,
    basic_df$Loser)))

basic_model_weighted <- BTm(cbind(Wsets,Lsets),W,L,id='player_',weights
    = w,data=basic_df)
summary(basic_model_weighted)

Abilities <- data.frame(BTabilities(basic_model_weighted))
Abilities[order(-Abilities$ability), ]

#visualization
library('qvcalc')
tennis.qv <- qvcalc(BTabilities(basic_model_weighted))
plot(tennis.qv)

### order effect (if there's a rank difference)
df_oe <- read.csv('Data_order_eff.csv')
df1 <- df_oe %>% select(home,away,Hsets,Asets)
df1$H <- factor(df1$home, levels=unique(c(df1$home, df1$away)))
df1$A <- factor(df1$away, levels=unique(c(df1$home, df1$away)))
head(df1)
Model1 <- BTm(cbind(Hsets,Asets),H,A,id='player_',data=df1)
summary(Model1)
df1$H <- data.frame(player_ = df1$H, at.home = 1)
df1$A<- data.frame(player_ = df1$A, at.home = 0)
df1
Model2<- update(Model1,formula = ~player_ + at.home)
summary(Model2)

### order effect (if the rank difference is larger than 50)
df1$HR = as.numeric(df_oe$HRank)
df1$AR = as.numeric(df_oe$ARank)
which(df1$AR - df1$HR < 50)

df1$H$at.home[which(df1$AR - df1$HR < 50)]=0
df1$H$at.home

Model3<- update(Model1,formula = ~player_ + at.home)
K = summary(Model3)


### order effect (increasing order effect for increasing rank difference
    )
```

```
df1$H$at.home[which(df1$AR - df1$HR < 75)] = 0
for (i in c(1:length(df1$H$at.home))){
  rank_diff = df1$AR[i] - df1$HR[i]
  print(rank_diff)
  if (is.na(rank_diff)==T){
    df1$H$at.home[i] = 0
  }
  else{if (rank_diff < 75){
    df1$H$at.home[i] = rank_diff/75
  }
  else{df1$H$at.home[i]=1}}
}

Model4<- update(Model1,formula = ~player_ + at.home)
K = summary(Model4)

Abilities <- data.frame(BTabilities(Model4))
Abilities_ordered_4 <-Abilities[order(-Abilities$ability), ]

### weighted order effect
df1$w <- df_oe$w
Model5<- update(Model1,formula = ~player_ + at.home,weights=w)

Abilities <- data.frame(BTabilities(Model5))
Abilities_ordered_5 <-Abilities[order(-Abilities$ability), ]
```

Simple Elo Model

⋮

```
# Read in the matches.csv file
matches <- read.csv("Data-2019.csv")

# Find the number of unique players in the data.
unique_players <- unique(c(matches$Winner, matches$Loser))
num_unique_players <- length(unique_players)

# Initialize ELO ratings for each player
players_elo <- rep(1600, length(unique_players))
names(players_elo) <- unique_players


# Define a function to calculate the expected outcome of a match
expected_outcome <- function(rating_a, rating_b) {
  return(1 / (1 + 10^((rating_b - rating_a)/400)))
}

# Define a function to update the ELO ratings after a match
update_elo <- function(winner_rating, loser_rating, k=32) {
  winner_expected_outcome <- expected_outcome(winner_rating, loser_
    rating)
  loser_expected_outcome <- expected_outcome(loser_rating, winner_rating
    )

  winner_new_rating <- winner_rating + k * (1 - winner_expected_outcome)
  loser_new_rating <- loser_rating + k * (0 - loser_expected_outcome)
```

```
25
26   return(c(winner_new_rating, loser_new_rating))
27 }
28
29 # Iterate over each match and update the ELO ratings
30 for (i in 1:nrow(matches)) {
31   winner <- matches$Winner[i]
32   loser <- matches$Loser[i]
33
34   new_ratings <- update_elo(players_elo[winner], players_elo[loser])
35
36   players_elo[winner] <- new_ratings[1]
37   players_elo[loser] <- new_ratings[2]
38 }
39
40 # Sort the players by their ELO ratings and store in a data frame
41 sorted_players <- data.frame(player = names(players_elo), elo = players_
     elo)
42 sorted_players <- sorted_players[order(sorted_players$elo, decreasing =
     TRUE), ]
43
44 # Write the sorted list to a file named "sorted_players.csv"
45 write.table(sorted_players, file = "sorted_players_with_ratings.csv",
     row.names = FALSE, sep = ",")
```

Complex Elo Model (with Match Characteristics)

⋮

```
1
2 surface_weights <- function(player_name, player_data) {
3
4   # Read the input csv file
5   #player_data <- read.csv(file_path)
6
7   # Filter the data for the given player name
8   player_data <- subset(player_data, player_data$Winner == player_name)
9   # print(player_data)
10  # Get the total number of surfaces played by the player
11  surfaces_played <- unique(player_data$Surface_x)
12
13  # Initialize a vector to store the weights for each surface
14  weights <- rep(0, length(surfaces_played))
15  # Calculate the sum of winning percentages for all surfaces played by
      the player
16  total_weight <- sum(player_data$winning_percent)
17
18  # Calculate the weight for each surface played by the player
19  for (i in 1:length(surfaces_played)) {
20    surface <- surfaces_played[i]
21    weight <- sum(player_data$winning_percent[player_data$Surface_x ==
     surface]) / total_weight
22    weights[i] <- weight
23  }
24
25 # print(weights / sum(weights))
```

```r
  # Return the normalized weights
 # return(weights / sum(weights))
  if ("Clay" %in% unique(player_data$Surface_x)) {
    Clay_Weight <- weights[which(unique(player_data$Surface_x) == "Clay")]
  } else {
    Clay_Weight <- 0
  }
  if ("Grass" %in% unique(player_data$Surface_x)) {
    Grass_Weight <- weights[which(unique(player_data$Surface_x) == "Grass")]
  } else {
    Grass_Weight <- 0
  }
  if ("Hard" %in% unique(player_data$Surface_x)) {
    Hard_Weight <- weights[which(unique(player_data$Surface_x) == "Hard")]
  } else {
    Hard_Weight <- 0
  }
  if ("Carpet" %in% unique(player_data$Surface_x)) {
    Carpet_Weight <- weights[which(unique(player_data$Surface_x) == "Carpet")]
  } else {
    Carpet_Weight <- 0
  }

  final_weights <- c(Hard_Weight, Grass_Weight, Clay_Weight, Carpet_Weight)
  min_val <- min(final_weights[final_weights > 0]) # find the non-zero positive minimum value
  final_weights_1 <- ifelse(final_weights <= 0, min_val, final_weights) # replace non-positive values with minimum
  final_weights <- final_weights_1

  if (surface == "Carpet") {
    k_surface <- final_weights[1]
  } else if (surface == "Hard") {
    k_surface <- final_weights[2]
  } else if (surface == "Grass") {
    k_surface <- final_weights[3]
  } else if (surface == "Clay") {
    k_surface <- final_weights[4]
  }

  return (k_surface)
}
get_k_factor <- function(surface, series, round, winner, player_data ) {
  k_base = 32

  if (series == "Grand Slam") {
    if (round == "1st Round") {
      k_tournament <- 0.2
    } else if (round == "2nd Round") {
      k_tournament <- 0.6
```

```
    } else if (round == "3rd Round") {
      k_tournament <- 0.8
    } else if (round == "4th Round") {
      k_tournament <- 1.6
    } else if (round == "Quarterfinals") {
      k_tournament <- 2.4
    } else if (round == "Semifinals") {
      k_tournament <- 3.2
    } else if (round == "Final") {
      k_tournament <- 4.0
    } else {
      k_tournament <- 1
    }
  } else {
    if (round == "1st Round") {
      k_tournament <- 0.2
    } else if (round == "2nd Round") {
      k_tournament <- 0.4
    } else if (round == "3rd Round") {
      k_tournament <- 0.5
    } else if (round == "Quarterfinals") {
      k_tournament <- 0.6
    } else if (round == "Semifinals") {
      k_tournament <- 0.8
    } else if (round == "The Final") {
      k_tournament <- 1
    } else {
      k_tournament <- 1
    }
  }

  # Example usage with an if condition to display weights for different surfaces
 # weights <- surface_weights(winner, file_path)
 # print(weights)

 k_surface <- surface_weights(winner, player_data)

  return(k_base * k_tournament * k_surface)
}

library(dplyr)

# Read in the matches.csv file
matches <- read.csv("New_Data_2019.csv")
player_data <- read.csv("Winning_Percent.csv")
#player_data <- read.table("Winning_Percent.csv", header = TRUE, sep = ",")
# Add a column for k factor
matches$k_factor <- mapply(get_k_factor, matches$Surface, matches$Series, matches$Round,matches$Winner, player_data )

# Find the number of unique players in the data.
unique_players <- unique(c(matches$Winner, matches$Loser))
num_unique_players <- length(unique_players)
```

```r

# Initialize ELO ratings for each player
players_elo <- rep(1600, length(unique_players))
names(players_elo) <- unique_players

# Initialize the number of matches played for each player
players_matches_played <- rep(0, length(unique_players))
names(players_matches_played) <- unique_players

# Define a function to calculate the expected outcome of a match
expected_outcome <- function(rating_a, rating_b) {
  return(1 / (1 + 10^((rating_b - rating_a)/400)))
}

# Define a function to update the ELO ratings after a match
update_elo <- function(winner_rating, loser_rating, k) {
  winner_expected_outcome <- expected_outcome(winner_rating, loser_
    rating)
  loser_expected_outcome <- expected_outcome(loser_rating, winner_rating
    )

  winner_new_rating <- winner_rating + k * (1 - winner_expected_outcome)
  loser_new_rating <- loser_rating + k * (0 - loser_expected_outcome)

  return(c(winner_new_rating, loser_new_rating))
}


# Iterate over each match and update the ELO ratings
for (i in 1:nrow(matches)) {
  winner <- matches$Winner[i]
  loser <- matches$Loser[i]
  surface <- matches$Surface[i]
  series <- matches$Series[i]
  round <- matches$Round[i]

  # Update the number of matches played for each player
  players_matches_played[winner] <- players_matches_played[winner] + 1
  players_matches_played[loser] <- players_matches_played[loser] + 1
  # matches_played <- length(players_matches_played[winner])
  # Get the k factor for the match
  k <- get_k_factor(surface, series, round , winner, player_data)

  new_ratings <- update_elo(players_elo[winner], players_elo[loser], k)

  players_elo[winner] <- new_ratings[1]
  players_elo[loser] <- new_ratings[2]
}

# Sort the players by their ELO ratings and store in a data frame
sorted_players <- data.frame(player = names(players_elo), elo = players_
    elo)
sorted_players <- sorted_players[order(sorted_players$elo, decreasing =
    TRUE), ]
# Write the sorted list to a file named "sorted_players.csv"
```

```
write.csv(sorted_players, file = "complex_sorted_players_with_ratings.
    csv", row.names = FALSE)
```

Model Validation

⋮

```
RANK <- read.csv('rank_2000_2019.csv')
colnames(RANK)<-c('date','rank','player')
rank_true <- RANK[which(RANK$date=='16/09/2019'),] #published rank at a
    certain day

rank_predicted <- Abilities_ordered_5
rank_predicted$players <- rownames(rank_predicted)
rank_predicted$rankings <- 1:nrow(rank_predicted)
rownames(rank_predicted)<-1:nrow(rank_predicted)

head(rank_predicted)

#match predicted rank for published top100 players
prediction = c()
for (i in c(1:100)){
  player = rank_true$player[i]
  predicted = rank_predicted$rankings[which(rank_predicted$players==
    player)]
  if (length(predicted)==0){
    prediction[i]=rank_true$rank[i]
  }
  else{
  prediction<-append(prediction,predicted)}
}

cor.test(rank_true$rank,prediction,method='kendall')
cor.test(rank_true$rank,prediction,method='spearman')
plot(rank_true$rank,prediction)
lines(c(1:100),c(1:100),col='red')
```