Predicting the Outcome of ATP Tennis Matches

Julian Bouchard
ax008241@acsmail.ucsd.edu
julian.bouchard@tum.de
University of California, San Diego

Eduardo Slonski
ax008293@acsmail.ucsd.edu
eduardoslonski@gmail.com
University of California, San Diego

ABSTRACT

The prediction of the outcome of a tennis match has always been a fascination of fans, sponsors, and the betting community alike. In this paper, we discuss the prediction of men's tennis matches. After exploring our dataset that includes matches from 2000 up until 2022, we perform informed feature selection and engineering. We then test and evaluate our model using various machine learning techniques. Our model with XGBoost performed the best, improving the 65.29% baseline model by 5.38% to achieve a final accuracy of 70.67%. We compare our results with already existing literature and ultimately embed our models in a web-based application where one can bet on tennis match outcomes against our models.

KEYWORDS

data mining, machine learning, supervised learning, sports betting, tennis

1 INTRODUCTION

In the world of sports, predicting the outcome of a match has always been a fascination of fans, sponsors and the betting community alike. The sports betting industry has grown massively in the United States over the past few years and is predicted to steadily increase further. The phenomenon has also gained significant traction in other parts of the world, such as Russia, and is becoming increasingly popular in East Asia [4]. All of this spells out a massive global market in the very near future which provides notable monetary and research motivation to accurately predict matches based on historical data.

Throughout this paper, we will be specifically discussing the prediction of Tennis matches as it is an individual versus individual sport. It therefore has significantly fewer variables as a massive team vs. team sport—such as soccer or football—where the teams, coaches and organizations change on a constant basis. We will train multiple prediction models, evaluate these and finally embed them in a web-based tennis match prediction game where the player tries to out-predict our models.

2 DATA

As noted in the introduction, we will be focusing on the sport of tennis. Specifically, we will explore men's professional tennis in the context of the Association of Tennis Professionals (ATP) [12].

2.1 Data Acquisition

Our data was sourced from a GitHub repository maintained by Jeff Sackmann [11]. This archive is continuously updated and contains data on all ATP matches going back to 1968. The repository encompasses multiple match types: singles, doubles, futures and challenger qualifiers. For the sake of this paper we will be limiting our scope to just the highest level of play—ATP singles—from 2000 to 2022. This equates to 68000 recorded matches.

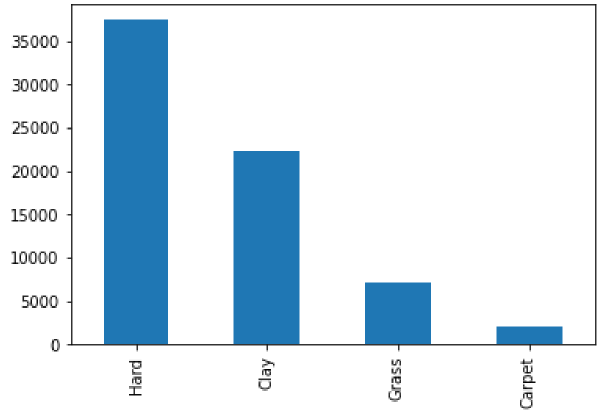
In total, each observation in the data frame has 49 features which encompass match information such as the players, their heights and ages, surface of the court, match result and much more. It also contains information about statistics during the match, like who won the first point. Since we are creating a model to predict future results, we will only use information that is possible to get before the match starts.

2.2 Exploratory Data Analysis

Taking a quick look at our data we notice that quite a few columns contain information that would be unavailable to us before the match. For example the final score of the match, who won the first point, how many aces each player performed, etc. Seeing as we are trying to predict the winner of the match before it starts, we cannot use those columns. We therefore drop 20 such columns before continuing our data analysis. We could then start our EDA, exploring the distributions of various features.

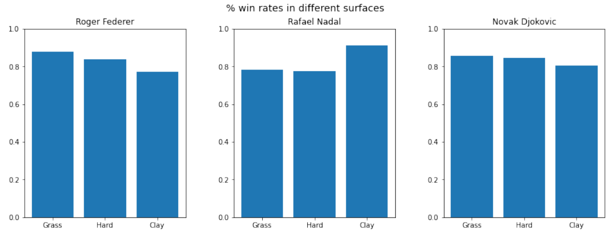
Court Surface.
We started by exploring how common certain court surfaces were in our data. The surface of the match is an important feature because it plays a big role in the player's style. In Fig.1 we can see that most of the matches were played on hard courts and clay, accounting for 86% of the total matches played.

Figure 1: Distribution of court surfaces



It is very interesting to see that players really do have different performances on different surfaces. Fig.2 shows the win rates of the Big 3 (Roger Federer, Rafael Nadal and Novak Djokovic) on different court surfaces. We can clearly see that Roger Federer prefers grass, Nadal dominates on clay—with an impressive 91%—and Djokovic is a more balanced player. The competitors themselves are also very much aware of this effect. Roger Federer is quoted saying “I grew up on clay, but grass is my favorite surface”[9]. Rafael Nadal said: "I think I am a complete player. I can play well on all the surfaces. For me, the clay might be easiest, but I am not a specialist on clay."[2]. Finally, Novak Djokovic: "Even though I grew up in Serbia mostly playing on clay, and I haven’t played on grass till I was 17 I think, first time,” he said. “Over the years I have a special relationship with the grass. I’m really happy that I was able to develop this kind of all-around success on all the conditions and tournaments.[5]. Their attitudes are reflected in their win rate statistics. With that in mind, we already know that the surface is going to play an important role in our model.

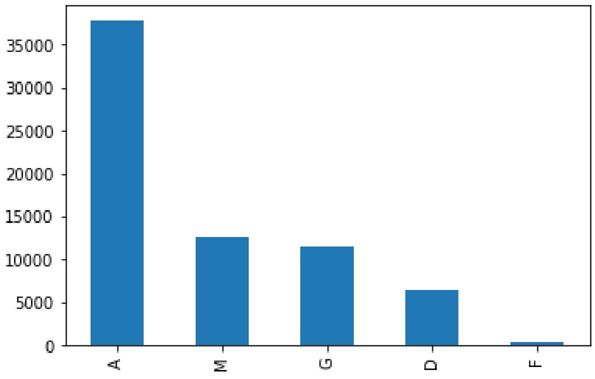
Figure 2: Win rates of the Big 3 on different surfaces



Tournament Level.

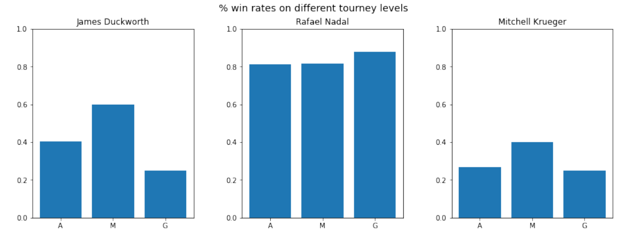
Next, we take a look at the tournament level feature. It represents within which type of tournament the match was played and is encoded in the following way: ’G’ = Grand Slams, ’M’ = Masters 1000s, ’A’ = other tour-level events, ’C’ = Challengers, ’S’ = Satellites/ITFs, ’F’ = Tour finals and other season-ending events, and ’D’ = Davis Cup. We learn from Fig.3 that most matches are played within the “other tour-level events” category, followed by Masters 1000s and Grand Slams which are almost equally present.

Figure 3: Distribution of tournament types



We also did an analysis on the win rate of players on different tournament levels. As we can see in Fig.4, players that are not in the top ranks (James Duckworth and Mitchell Krueger) tend to have even lower win rates on G (Grand Slams). That is expected, because those tournaments are harder and more competitive.

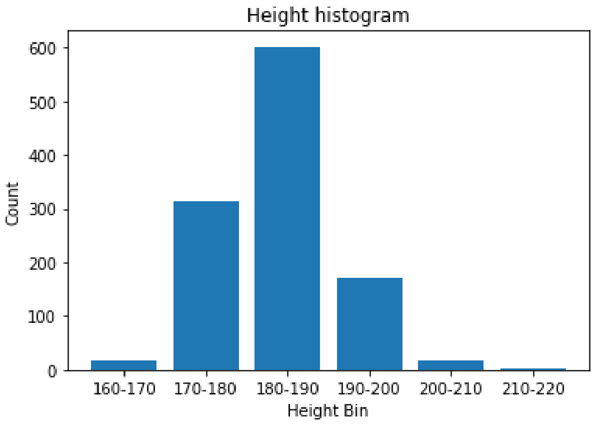
Figure 4: Comparing players across different tournament levels



Player Heights.

Moving on to player heights, Fig.5 depicts the histogram of all heights in our dataset. It shows a typical normal distribution, just like the overall height distribution of the population. Unlike the general population, however, we notice an above average median height.

Figure 5: Distribution of player heights



Investigating how heights might impact win rate, we created Fig.6. The win rate by height shows that there exists a trend that tall players tend to win more. This can be rationalized with game knowledge. The arms of the tall players are naturally higher above the net, making it easier to perform great servers. However, we can also see in the graph that the trend is not very significant. We will use this feature in the model, yet we already suspect that it might not be as relevant.

Player Rank.

Lastly, we analyze the available rank data. We already expect that the rank of the player is a strong indicator of their capabilities. Fig.7 plots the win rates using bins of 50. A downward curve is to be expected since low ranked players are intuitively more likely to lose more than high ranked players. Nonetheless, it is interesting to observe just how large the difference between the first 50 and second 50 ranked players is. The following decline is much smaller. Zooming into this first bin results in Fig.8. Even within the top 50 players we see a clear downward trend in win rate as we go down the leaderboard.

Figure 6: Win rate by player height

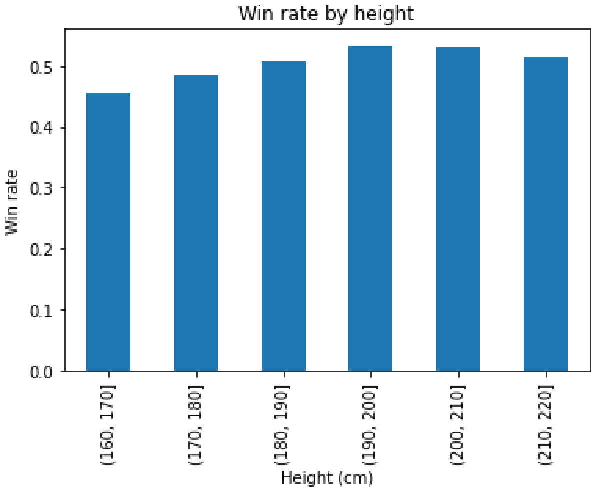
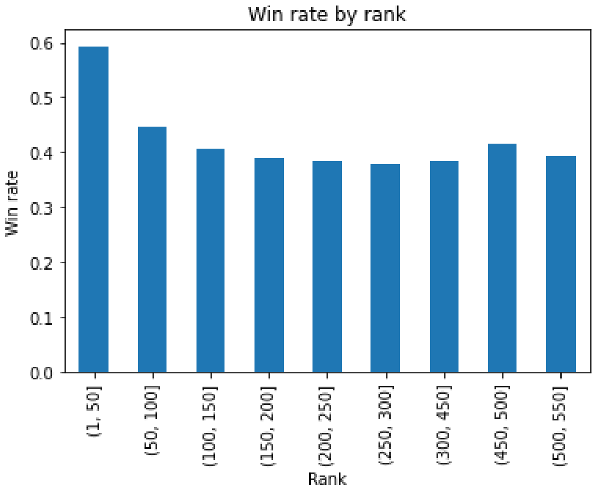


Figure 7: Win rate by rank distribution of the top 550 players



Interestingly, the ranks past 100 do not exhibit the same trend as the first 100. Fig.9 depicts an equal win rate no matter the ranking between 91 and 150. This analysis implies that the rank is an important feature, but it is way more important for high ranked players than low ranked ones. We will use the rank as a raw feature, to not extend the complexity of the model too much. But we encourage new studies to take this difference into account.

2.3 Feature Engineering

Wins and Losses. The columns that we add to our dataframe are the total wins and losses of each player up to that match. This also allows us to calculate the total matches that a player has played by that point in time. In turn, we can then add the win rates of each player.

Figure 8: Win rate by rank the top 50 players

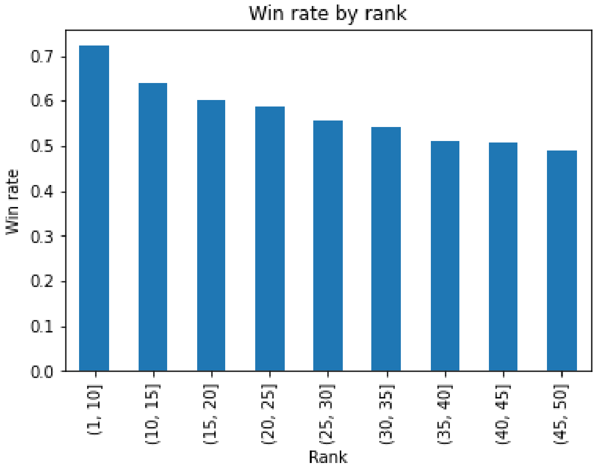
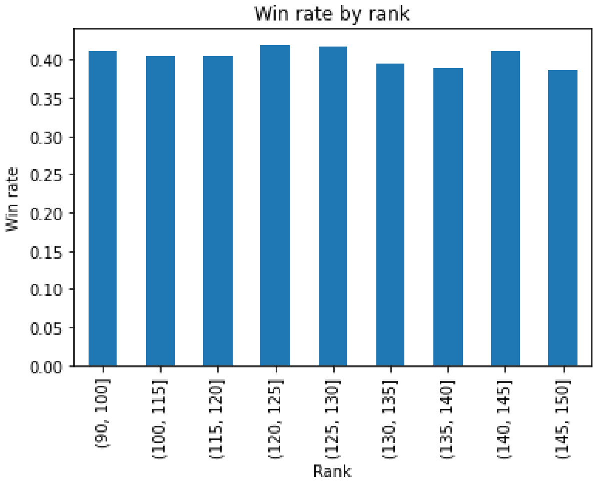


Figure 9: Win rate by rank of the top 100 - 150 players



Surface. Each player has their preferred court surface. In some cases, a higher ranked player might lose to a significantly lower ranked player depending on the surface. Therefore, to get a better understanding of this phenomenon, we calculated the wins and losses of each player for the specific surface of the court that is being played on.

Player Specific Matchups. We also want to know how well a player has performed historically against this specific opponent. For this we added some features that encapsulate the amount of wins and losses of head to head matches between player A and player B up until this point in time.

Recent Performance. Additionally, to capture a player's current momentum, we calculated the wins and losses for said player within a time window of 6 months. With this information we will know if a player has significantly improved recently, if they are consistent or if they fell off.

Performance given Tournament Level. Finally, we wish to know how well a player does at different tournament levels. A small number of wins in the Grand Slams, for example, are much more valuable than many wins in a lower level. To describe this we calculate the wins and losses up to this match of each player for the specific tournament level that is being played.

Dataset Organization.
A big problem is that we cannot use the entire dataset to calculate those features for each match. This is because we would train our model on information about the future. For example, if we have a match Rafael Nadal vs Roger Federer, played in 2015, we would be calculating the feature 'head to head' for that match using data from after that match. That would be cheating and at the end of the day, the model would not generalize well to real, new data. Instead, we had to ensure that we calculate each feature only with the data that preceded the specific match.

A hurdle that we quickly ran into is that the original dataset was not ordered in a particular way and the match ids could not be used to determine the proper sequential ordering. However, the dataset did contain a 'tourney_date' column. The problem with this is that 'tourney_date' specifies the date that the tourney started, but not necessarily the match. This was overcome by using the 'match_num' column which specifies what number this particular match was within its respective tournament. We combined the two columns and with that we could order the dataset by date.

We created a base algorithm that can iterate each row and select the information of all previous rows, matching the selected filter for each feature. Running that ended up taking a lot of time, so we created another algorithm that was more runtime efficient (90% faster), but it was unstable due to RAM constraints, since the dataset is large and it needed to calculate and store a large amount of data for each row. Therefore, we decided to use the first approach, which was slower, but more stable.

Preparing the data for the models.
We need to address the problem that the dataset provides 'winner_name' and 'loser_name', but not the result itself. For that, we created a new dataset that used the information of the loser and the winner to create three new columns: 'player1_name', 'player2_name' and 'result'.

To accomplish that, our first approach was to duplicate the dataset, inverting the names and the results. For example: we have 'winner_name' = 'Rafael Nadal', and 'loser_name' = 'Roger Federer', first we will change to 'player1_name' and 'player2_name', and create the column 'result' = 1. After that we invert the players: now player 1 = 'Roger Federer' and player 2 = 'Rafael Nadal' and also invert the result, 'result' = 0. We need to do that so the model will be able to train to predict wins and losses, according to the features of player1 and player2.

A problem with this approach is that we end up with a lot of bias in the dataset itself, since we are duplicating it in its entirety. So we had another idea: we first split the dataset randomly in two, then we use the first split to be the winners and the second split to be the losers. After that we just concatenate the two datasets. We tested both cases and observed almost no difference in accuracy, but the second approach was slightly higher so we decided to use that for the final model.

Post preliminary model features.
We ran some initial models using the above features alongside original features like player rank, heights and ages, but these models were not performing at a satisfactory level. Eventually we came up with an idea that boosted our performance. The problem with the original model was that it was learning to understand if a player had a high win rate, it was more likely to win, for example. But what would happen if we had two players with high win rates playing against each other? Let's say 65% and 67% of win rates. The initial model naturally would think that both are likely to win, because they have higher win rates than a 55% player for example. But in reality the second player is more likely to win in this case. With that in mind we created a new feature: the difference between two players. We created a new set of features that are the calculated difference between the win rates, in that case it would be 2% (67 - 65) and every other numerical feature. That addition alone made our model jump from 67.8% of accuracy to 69.5%, almost 2% with a fine tuned feature. We also kept the original, non-differenced features to fully capture certain dynamics of the game. A 2% difference for high ranked players might contribute differently to the match outcome than the same difference for low ranked players.

2.4 Final Data

All in all, our final dataset uses 44 features with the following breakdown:

- 2 for the players
- 4 for the general wins (2 win_rates and 2 total matches)
- 4 for the surface (2 win_rates and 2 total matches)
- 4 for the form 'last 6 months' (2 win_rates and 2 total matches)
- 16 for the tourney levels (2 win_rates and 2 total matches, times 4 different levels)
- 2 players heights
- 2 players age
- 2 rank
- 2 rank_points
- 6 for differences between the two players

3 PREDICTIVE TASK

Our goal is to predict the winner of a given match based on all historical data up until that match. We are not predicting based on future matches.

We evaluate all of our models using the accuracy score. This is most appropriate because we simply care about the percentage of how many matches our model can predict correctly.

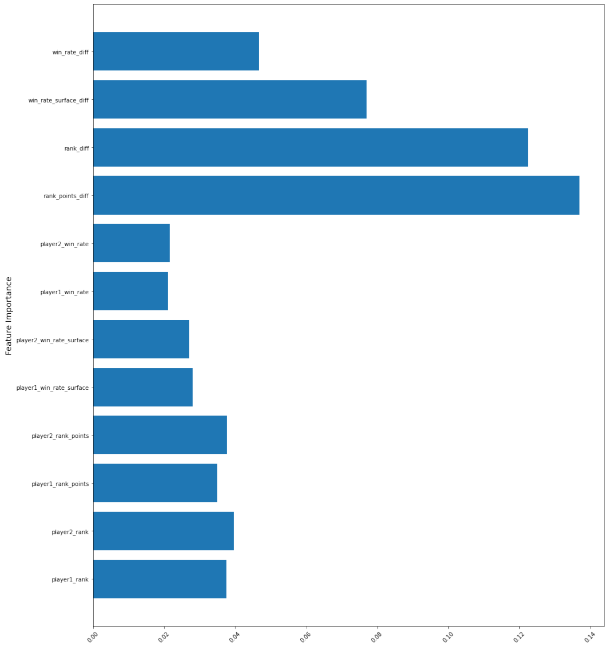
We will be comparing our models to two baselines. The first baseline is a simple coin flip model. None of our models should perform worse than a 50/50 guess. This represents the absolute minimum that our models need to beat to even be considered. For the second baseline, we always predict that the higher ranked player will win. This provides a solid baseline as one would assume that on average a higher ranked player beats the longer ranked player. This baseline performs quite well, achieving a 65.29% accuracy on our dataset. That demonstrates, in essence, that the rank system in tennis makes sense. We are aiming to get 5% more than the baseline. That can be hard, due to the randomness of the game and because the rank itself is already a measure of winning. A great explanation of how the system works can be found here [10] and here [3]. That being said, we believe that our models will beat this baseline as it does not take any other factors into account. Court surface, for instance, can have a significant impact on who wins. The baseline model also does not have information about the momentum of a player at a given time. In the cases of rising (or falling) stars, the baseline will almost always predict wrong.

4 MODELS

With our prepared data and clear goal in mind, we trained numerous models with various hyperparameters: Logistic Regression, Random Forest, XGBoost, Gaussian Naive Bayes, LGBM and Ridge Classifier (we also tested other algorithms but these were our main focus). The best out of the box performers were Random Forest, LBM and XGBoost. We then performed grid search on all three of them and determined that our best performing model in terms of accuracy is XGBoost with 200 estimators.

We also used the Random Forest model to analyze what features it found most important. The 12 most important features are shown in Fig.10. It is interesting to see how the features that measure the difference between the athletes play a big role in the model, significantly more so than the raw values themselves. It is, however, important to note that for the Random Forest model it is common that when we have highly related features, one of them will 'cannibalize' the other which might rationalize the contrast in perceived importance.

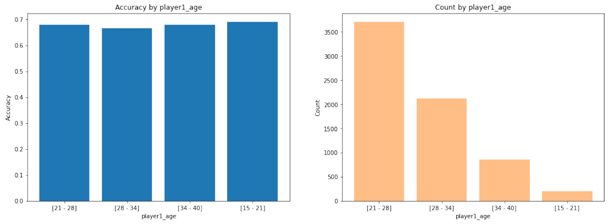
Figure 10: Feature importance of the top 12 features in Random Forest



Bias Evaluation.

We also checked if the models were performing worse on certain groups for each feature, for example getting a high accuracy for taller players and a lower accuracy for short players, but it this was not the case. The model generalized well in that sense, mostly because the values are generally not sparse. For the few that were (like ranks and rank points), we used some pre-processing techniques to address the issue. This mostly consisted of taking the log instead of using the raw values. Fig.11 shows the accuracy of our Random Forest model based on age groups next to the age group distribution in the data.

Figure 11: Random Forest accuracy by age group compared to the age group distribution



5 RESULTS

Model	Accuracy
Baseline	65.29%
Logistic Regression	67.97%
Gaussian Naive Bayes	67.12%
Random Forest	68.81%
XGB	70.67%
LGBM	69.81%
Ridge	68.15%

Table 1: Table containing all tested models accuracies

All of our models outperform both baselines. Our best model boasts a noteworthy increase in accuracy compared to the second baseline model. It achieved a final accuracy score of roughly 70.67%. Given the context, this is impressive and on par with models we have seen from other sources [7],[6],[8].

Our model can outperform the baseline because it includes information about external factors—such as the court surface type—as well as certain player attributes—like height, age and whether they are right handed or not. Additionally, a substantial model improvement was made when we included features relating to player vs player history. Knowing player X consistently beats player Y no matter their current ranking can be a key factor when making a prediction. Lastly, like shown in Fig.10, including the difference in rank instead of just the raw ranking can lead to better predictions by our model.

That being said, not all features are equally important. Taking a look back at Fig.10 we can tell that a player's physical attributes, like height and age, are not one of the 12 most important features. In general, we found them to be of very low priority. This confirms our hypothesis from our EDA.

6 LITERATURE

Predicting the outcome of tennis matches before they happen is by no means a novel task. Various research papers and dissertations have been written on this topic. The prominent ones that we focused on were (1) a capstone project by Nicholas Devin [8], (2) a Master's dissertation by Alexander De Seranno [7], (3) a research paper by Jack C. Yue et al. [13] and lastly, (4) a paper by Cornman, Spellman and Wright [6].

All of the above, including ourselves, sourced the data from Jeff Sackmann's GitHub repository. However, (1) only used the Women's Tennis Association (WTA) datasets and (3) focused only on ATP Grand Slam matches. Whereas, (2), (4) and ourselves used the ATP Singles datasets. However, (2) differentiates itself by also using Futures and Challengers to assess players that only recently entered the Tour level tournaments. Additionally, the dataset used in (4) is also slightly divergent to the set we used. They merged the original dataset with the sports betting odds at the time. This caused them to lose 7% of the matches. In summary, the datasets employed by (2) and (4) are most in line with ours.

In terms of approach, (1) and (4) were the most similar to our work. Both research papers performed akin feature engineering—such as features relating to head-to-head matchups—and tested almost the same algorithms. (2) used a slightly alternate approach and utilized a Neural Network in their model. Lastly, (3) opted for the most dissimilar approach: They used a Glicko model to rank players. They performed quite well, resulting in the highest accuracy out of all four papers. It is interesting and important to note that all approaches—including our own—use a model based solely on rank as their baseline.

What makes our model stand apart from all of the aforementioned approaches is the fact that for each match prediction we only use the data that is historically available at that point in time. Moreover, even though the model of (4) also uses differences between the two players in their features, they only use differenced features whereas we purposely included the raw counterparts (i.e. true rank, rank points etc.) to better capture the situation.

Comparing our results to the existing literature, we observe that our final models perform as well as, if not better than, the models proposed in (1), (2), and (4). The only approach that boasts a notably better score is that of (3) where the Glicko model is applied. That being said, our improvement of 5.38% over our baseline model is the largest increase in margin reported in any paper.

7 WEB-APP DEMONSTRATION

7.1 Idea

We wanted to make the demonstration of our models fun and interactive. Therefore we created a web-based game where the player and the models attempt to predict the outcome of a given match. Both are given statistics to aid in their decision. In order to make the game more interesting than just predicting who the winner is, we decided to let both parties bet with their level of certainty. In other words, if the player is 80% certain that X wins in the Y vs X matchup, and the model predicts 60% for X, if X actually won that matchup then the player gains points. Otherwise, if X lost then the model was closer and the player loses points. Multiple rounds are played until either the player reaches 0 and loses or they reach 100 points and win. In addition to this, the player can select a difficulty before starting a game. The selected difficulty corresponds to a different model. Easy is Logistic Regression, medium is Random Forest and hard is XGBoost all with increasingly higher accuracy.

7.2 Implementation

The demonstration was implemented in Typescript using the React library to control states and the MaterialUI library for the visual components. The app was designed as mobile-first and works on all devices. A production build of the app can be run locally by cloning the GitHub repository at https://github.com/JulianBouchard/DSC148-Project and running yarn install followed by yarn start in the /Frontend directory. This requires yarn [1] to be installed.

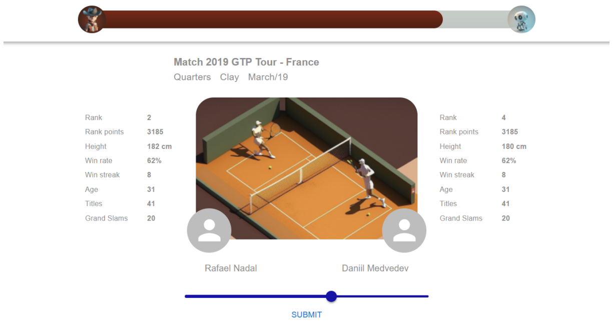
7.3 Final Product

When the app is launched a difficulty select overlay masks the screen. Here the user simply slicks one of the three options.

At the top of the main game screen is a score bar that represents the players points (1). Once the bar reaches either end the game will end showing either a win or loss screen. Right below that is the title which contains basic information about the context of the match (2). On either side of the screen are statistics belonging to their respective tennis players (3). At the bottom of the screen is a slider that the player uses to bet (4). A bet of 30% would mean that the player is 70% confident that the tennis pro on the left will win. On the other hand, a bet of 80% would signal 80% confidence in a win for the tennis pro on the right. After clicking the submit button (5) the score increases or decreases based on whether the player or computer was closer to the true answer. The amount by which the score changes is based on the difference in confidence between the player and the model prediction.

Once the player has won or lost, an overlay appears stating the outcome and allowing the player to choose a new difficulty and play again.

Figure 12: A screenshot of the main game screen



Match 2019 GTP Tour - France

Quarters Clay March/19

Rank	2
Rank points	3185
Height	182 cm
Win rate	62%
Win streak	8
Age	31
Titles	41
Grand Slams	20

Rank	4
Rank points	3185
Height	180 cm
Win rate	62%
Win streak	8
Age	31
Titles	41
Grand Slams	20

8 CONCLUSION

By combining Jeff Sackman's ATP Singles datasets from 2000-2022, we obtained 68000 observations of raw data across 49 variables. After analyzing this data and making informed decisions about which features to keep, we engineered many new features to allow our model to make more accurate predictions. This prominently included statistical differences, surface and head-to-head features. Our Random Forest model found the statistical differences to be quite important during its fitting. The most accuracy was gained using the XGBoost algorithm: 70.67%. This was a 5.38% improvement over our baseline model and performs at the same level or slightly above most other models present in various literature.

9 FUTURE RESEARCH

Throughout the course of this work we had a lot more ideas and interesting thoughts that we unfortunately did not realize due to time and complexity constraints and would like to share here.

We addressed many problems during the feature engineering and data preparation, but for further analysis it can be even more refined by removing rows for players that didn't play more than 100 matches in total, remove matches that were not completed (like in [7]) and remove rows for atypical times—such as during the pandemic.

Moreover, one could think of cutting the first 20 matches of every player from the dataset, and train and test using only the rows left. The problem with our "only using data up until this match" methodology is that we often had a problem with the first matches in our dataset as we do not have sufficient historical information to go on. This resulted in the first rows often containing statistics like 0 wins, 0 losses etc.

We were intrigued by [13]s use of the Glicko model and would have liked to implement this in our own model to directly compare the two.

Additionally, one could use a better approach to handle the 'head to head' scenarios. Most of the time the two players didn't play a lot of matches against each other which could lead to skewed data. In order to combat this, we could first cluster the players in n groups that have certain similarities. Then we could use the cluster information, instead of the exact player, to better estimate the 'head to head' or 'cluster to cluster'.

Another idea was to use data for the specific tournament that is being played. If the player is winning a lot in a specific tournament, it is more likely that he will continue to win there.

Lastly, we contemplated the idea of adding a feature that indicates whether the match is taking place in a player's home country or home stadium. The location, climate and crowd origin of the match can all affect a player's mental state.

REFERENCES

[1] [n. d.]. Fast, reliable, and secure dependency management. https://classic.yarnpkg.com/en/

[2] [n. d.]. Rafael Nadal quote. https://www.azquotes.com/quote/210986

[3] [n. d.]. Rankings: Pepperstone ATP Rankings FAQ. https://www.atptour.com/en/rankings/rankings-faq

[4] [n. d.]. Sports betting market size amp; share analysis report, 2030. https://www.grandviewresearch.com/industry-analysis/sports-betting-market-report

[5] [n. d.]. Tennis now - tennis news, tennis blogs, tennis forums, live scores, player profiles, Tennis Community, social network, TV schedule. https://www.tennisnow.com/Blogs/NET-POSTS/November-2022-(1)/Djokovic-s-Ability-to-Adapt-to-All-Surfaces-and-Si.aspx

[6] Andre Cornman, Grant Spellman, and Daniel Wright. 2017. Machine learning for professional tennis match prediction and betting. *Stanford Unverisity* 1, 2 (2017), 4.

[7] Alexander De Seranno. 2020. *Predicting Tennis Matches Using Machine Learning*. Master's thesis. Ghent University.

[8] Nicholas Devin. [n. d.]. Utilizing data to predict winners of tennis matches. https://nycdatascience.com/blog/student-works/utilizing-data-to-predict-winners-of-tennis-matches/

[9] Jovica Ilic. 2021. Roger Federer recalls: 'I grew up on clay, but grass is my favorite surface'. https://www.tennisworldusa.org/tennis/news/Roger_Federer/106620/roger-federer-recalls-i-grew-up-on-clay-but-grass-is-my-favorite-surface-/

[10] Utathya Nag. 2022. Tennis rankings: Everything you need to know. https://olympics.com/en/news/tennis-rankings-atp-wta-men-women-doubles-singles-system-grand-slam-olympics

[11] Jeff Sackmann. [n. d.]. ATP Tennis Rankings, results, and stats. https://github.com/JeffSackmann/tennis_atp

[12] ATP Tour. [n. d.]. Official site of men's professional tennis: ATP tour: Tennis. https://www.atptour.com/en/

[13] Jack C Yue, Elizabeth P Chou, Ming-Hui Hsieh, and Li-Chen Hsiao. 2022. A study of forecasting tennis matches via the Glicko model. *Plos one* 17, 4 (2022), e0266838.