The Risk and the Reward: An In-depth Evaluation of Strategic Aggressiveness in Men's Tennis

Abstract

In a tennis match, when do players know the right situation to be aggressive? The role of aggressiveness on in-game tennis strategy is one of the most overlooked aspects of the sport. Using shot-by-shot data, we seek to answer the following question: holding ranking constant, based on the in-game situation, what level of aggressiveness is most strategic and yields the largest reward?

1 Setting the scene

In the 2008 Wimbledon final, one of the most iconic tennis matches in history, Rafael Nadal faced Roger Federer in a battle that tested every element of their game. As rain delays stretched the match to nearly five hours, both players adapted to the grueling conditions. What ultimately tipped the balance in Nadal's favor, allowing him to seize victory after years of falling short at the All England Club, was not just his consistency, but his tactical aggression in key moments. In the fifth set, with the pressure at its peak, Nadal began stepping into the court, taking the ball earlier, and attacking Federer's backhand. His aggressive mindset, honed through years of adapting to pressure situations, enabled him to seize control and clinch his first Wimbledon title [1].

Aggressiveness in tennis is more than just raw power; it is a carefully calibrated response to various match conditions. Understanding when to attack and when to play it safe can be the difference between victory and defeat. But how do players determine the right level of aggressiveness to employ? By quantifying this elusive quality through an aggressiveness metric, players can develop a deeper understanding of how their choices in different scenarios impact the flow and outcome of a match. This paper explores how players can use the context of the match to determine how aggressive they should be.

2 Data

2.1 Description

The data we used is from Jeff Sackmann's Match Charting Project. The [dataset] offers a detailed account of every point during a given tennis match, tracking the progression of the score and capturing key events such as serves, rallies, and point outcomes. At a high level, the data provides insights into player performance across various stages of a match, highlighting serve effectiveness, rally dynamics, and match context [2]. This enables an analysis of player strategies, decision-making, and execution under competitive conditions, as well as the factors that contribute to winning or losing individual points. This data was charted for hundreds of high-profile games across two decades. However, we chose to focus on data from **2020 to 2023**.

2.2 Rally Codes

The dataset contains several variables providing context to the game. The main feature is the comprehensive **shot-by-shot** codes embedded within the '1st' and '2nd' columns (first and second serve respectively). In it's original form, each row has a column with a sequence of numbers and characters which represent what happened in a given rally. These codes cover various aspects of the match, including serve zones, shot types, shot directions, and outcomes. By capturing this detailed information, the codes allow for a thorough analysis of each player's tactical decisions and performance across the match. This level of precision is essential for understanding the dynamics of the game, evaluating how players adapt to different situations, and assessing the impact of specific shots on the overall outcome of the match.

2.3 Pre-Metric Data Wrangling

The data wrangling process involved first extrapolating all of the points from point-by-point data to shot-by-shot data. By defining columns to collect each potential action that a shot could possess, we were able to create our model. We also created many new columns to help keep track both the of both the action-state of the game and the outcome of each shot. This included splitting up the shot code into different parts and creating variables that described shots in the rally. This allowed us to more easily define what an aggressive action looked like.

3 Creating an Aggressiveness Metric

3.1 Rating System

Our first goal was to determine the aggressiveness of a given shot. To do this, we qualitatively determined the level of aggressiveness the shot warranted based on the dimensions the code offered. We split our data into rallies, serves, and returns. Within each category, we created separate aggressiveness variables for different features such as shot type and shot location. These variables were then aggregated into one final aggressiveness metric.

For **rallies**, the `outcome_shottype_aggressiveness` is determined by evaluating the type of shot (`shot_type`). For example, offensive shots like overheads (`"o"`), volleys (`"p"`), and smashes (`"u"`) are assigned a high aggressiveness score of two, while more neutral shots like slices (`"j"`) and drop shots (`"k"`) receive a score of 1.5. conservative shots like lobs (`"r"`) and defensive slices (`"s"`) are assigned a score of zero. The `direction_aggressiveness` is calculated by comparing the direction of the shot (`direction`) and the next shot's direction (`next_direction`). A score of two is given if the shot significantly changes the direction, such as from one corner to the opposite corner (e.g., direction 1 to 3 or vice versa). If the next shot's direction is neutral or central (e.g., direction 2), it receives a score of zero. The `approach_aggressiveness` assesses whether the player is approaching the net (`approach == "+"`), with a score of two indicating an aggressive approach. These factors are combined into an overall `agg_rating`. Shot type and approach aggressiveness were both given a weight of two while directional aggressiveness was given a weight of one. An overhead (`"o"`) or smash (`"u"`) was automatically set to the maximum aggressiveness rating of ten.

For **serves**, the code calculates `serve_aggressiveness` based on the serve zone. For instance, a first serve into the wide serve zone (zone 6) is considered highly aggressive and assigned a score of two, while a serve to the body (zone 5) receives a score of zero. Additionally, `outcome_aggressiveness` evaluates the result of the serve, such as an ace (`"*"`) or an unreturned serve (`"#"`), and assigns a high aggressiveness score of two. The overall `agg_rating` for serves is calculated by combining the `serve_aggressiveness` and `outcome_aggressiveness` and multiplying the result by two. Aces and unreturned serves were given the maximum aggressiveness of ten.

Similar to rallies, the aggressiveness of a **return** shot is assessed by `outcome_shottype_aggressiveness`. For example, a return that is an aggressive shot type like an overhead (`"o"`) is given a high score of two. The `direction_aggressiveness` is calculated similarly, with a high score for returns that change direction significantly. Additionally, `depth_aggressiveness` is introduced, assigning a score of two for deep returns that land near the baseline (depth 9) and zero for shallower returns (depth 7). These measures are combined into a final `agg_rating` for returns. Depth and shot type aggressiveness were given weights of two while direction was once again given a weight of one. The rallies are also capped at a maximum value of ten for highly aggressive shots like overheads or smashes.

3.2 Practical application

In this project, we used K-means clustering to categorize tennis players into distinct tiers based on their aggressiveness score [3]. Initially, we filtered players who had played fewer than 1000 shots, a threshold determined using the Elbow Method [4]. This ensured that only players with enough data points were considered for reliable clustering. To find the optimal number of clusters, we used both the Elbow Method and Silhouette Score, ultimately selecting four clusters [5].

- **Ultra Aggressors**: These players showcase the most aggressive shot-making, frequently taking risks to dictate points. Notable names include Alexander Zverev (5.07), Stefanos Tsitsipas (5.05), and Nick Kyrgios (5.16). Their high aggression scores reflect a bold and assertive style of play, often pushing for winners early in the rally.
- **Moderate Aggressors**: Players in this tier balance aggression with strategic restraint, aiming to win points through a mix of assertive shots and consistent play. Carlos Alcaraz (4.98), Novak Djokovic (5.00), and Daniil Medvedev (5.00) are key figures, blending controlled offense with efficient point construction.

- **Balanced Players**: These players represent a balance between aggression and defense, often adapting their strategies based on match dynamics. Andy Murray (4.89), Casper Ruud (4.84), and Dominic Thiem (4.89) are prominent names, showing versatility and patience in their playstyle while still seizing opportunities for attack.
- **Conservative Players**: In this tier, players are more focused on defense, waiting for their opponents to make errors rather than forcing the issue themselves. Daniel Evans (4.59) and Max Purcell (4.54) exemplify this approach, relying on consistency and resilience rather than overt aggression.

This clustering process helped categorize player styles meaningfully, allowing for further analysis of strategies based on their aggressiveness tendencies.

4 Pre-Model Data Wrangling

4.1 Environment

Before we begin running our model, we determined which variables would be important in contextualizing aggressiveness.

4.1.1 Score

The score of the game should be accounted for. Based on the situation, different levels of aggressiveness might prove advantageous. Score is composed of three variables: **points**, **games**, and **sets**. While we were given all of this data, it still needed to be formatted. Our final goal is to get the score difference at each level (point, game, set) of the match and the total.

Points might pose some issues as it does not increase by one and does not increase linearly. Because of this, we decided to modify the values. We assigned *15* a value of one, *30* a value of two, *40* a value of three, and *AD* a value of four. We then created two variables - one for point differential from the hitter's perspective and one for the total number of points in the game. We did this for the game and set level as well.

However, due to occurrences such as the "advantage set" [6] or extremely long tiebreakers, all of our values had fairly long ranges. We anticipated that this would throw the model off. As a solution, we decided to categorize the differential and totals at each level into different bins that represent a specific situation.

Let's start with the **point level**. Using the calculated point differential and total points in the game, we created **seven** unique bins:

Bin #	**Description**
1	Player hitting the shot is **winning** a close game (one shot difference), but they are more than one shot away from winning.
2	Player hitting the shot is **losing** a close game, but they are more than one shot away from losing.
3	Player hitting the shot is **tied**.
4	Player hitting the shot is **winning** a close game and is one shot away from winning.
5	Player hitting the shot is **losing** a close game and is one shot away from losing.
6	Player hitting the shot is **winning** the game and it is **not close** (absolute difference greater than one).
7	Player hitting the shot is **losing** the game and it is **not close** (absolute difference greater than one).

Next is the **game level**. Similar to the point level, we used game differential and total games to categorize the game situation into **six** unique bins:

Bin #	Description
1	Player hitting the shot is **winning** a close set (one game difference), but is in a high pressure situation (more than eight games played in the set).
2	Player hitting the shot is **losing** a close set, but is in a high pressure situation.
3	Player hitting the shot is **tied** in a close set and is in a high pressure situation.
4	The game is close, but the set situation is pretty low pressure (eight games or less - no player has an advantage in the set).
5	Player hitting the shot is either **winning** the set by two games or **losing** by two games.
6	Player hitting the shot is either getting blown out in the set or is blowing out their opponent in the set.

And lastly, the **set level**. We used the set differential and total sets to categorize the values into **four** bins. We categorized such that it did not matter if this match was best of three or best of five sets.

Bin #	Description
1	Player hitting the shot is **winning** and the match is close (one set difference).
2	Player hitting the shot is **losing** and the match is close.
3	The match is tied on the set level
4	The match is not close on the set level (absolute difference greater than one).

Each of the levels will become its own variable for the model. We have successfully converted six variables with large intervals into three variables with discrete, categorized values. This will make it easier for the model to make a confident decision.

4.1.2 Ranking

The hitter and opponent's *ATP ranking* could be a potential confounding variable in our model. After all, better players are likely to have more success regardless of their aggressiveness. To account for this, we scraped the last few years of top 150 ATP rankings from ESPN and merged it with our dataset based on the year of the tournament [7]. We were able to get the ranking of the player hitting the shot and the opposing player. If the player was not in the ATP ranking dataset, we would know that they were unranked for that given year.

Similar to the *score* variable, we decided to categorize the rankings into bins. While we could have just inputted the ranking variables as their numeric values into the model, the large range would mean that some numbers would show up more than others and some might not show up at all. For both hitting player and opposing player rank, we categorized the values into **five** unique bins:

Bin #	Description
1	**Top 10** players.
2	Players ranked from **11-32** (top 32 players are seeded at grand slam tournaments).
3	Players ranked from **33-104** (top 104 in the world are given entry into a grand slam tournament).
4	Players ranked from **105-150**.
5	Players who are **unranked**

4.1.3 Putting it together

The score and rankings together give us the *game environment*. In other words, it gives context to the aggressiveness of a player for a given shot. Taking points, games, sets, and player rankings into account, there are $7 \times 6 \times 4 \times 5 \times 5 = \mathbf{4200}$ possible environments. And each environment gets paired with an aggressiveness level. Since aggressiveness is measured on a scale of 0-10 inclusive, there are $7 \times 6 \times 4 \times 5 \times 5 \times 11 = \mathbf{46200}$ possible environment-action pairs. This gives a good idea of the diversity that we are dealing with. This will become important when we try to determine the optimal aggressiveness for the different environments.

4.2 Reward

Now that the environments and actions have been properly defined, we need to create a "reward" metric. This will serve as the outcome/label variable in the model and provide a basis for determining how desirable a shot was.

How can we determine if a shot is reliable? Well if the player ends up winning the point, they should be rewarded for their shot selection. However, what happens if they hit a certain shot and end up winning the point five shots later? Obviously proximity to the actual point-winning shot should matter, but tennis is a tactical sport where past shots in the point can have an impact on the eventual outcome. Balancing between these two sides serves as the basis for our reward metric. For a player who eventually **won** the point, let

$$v_W = 10 \times 0.5^{\lambda_W}$$

where v_W is the final reward, and λ_W is the number of shots the player has left until they win the point. 10 is the maximum reward a player can receive which occurs on the shot where they win the point. If a player had two shots left to play before they won the point, for example, their reward would be **2.5**. For a player who eventually **lost** the point, let

$$v_L = -5 \times 0.5^{\lambda_L}$$

where v_L is the final reward (or punishment), and λ_L is the number of shots the player has left until they lose the point. -5 is the minimum score a player can receive which occurs on the shot where they lose the point. For example, if a player had two shots left to play before they lost the point, their reward would be **-1.25**.

Our method isn't perfect, mainly because our dataset did not distinguish between forced and unforced errors. We operated under the assumption that if a player hit a shot that led to a win or a loss, their shot influenced the result rather than the opponent's response.

With the reward function created and the environmental variables properly transformed, it is time to create the model.

5 Model creation

The chosen technique is an extreme gradient boosting (XGBoost) model. In XGBoost, a series of weak predictive models, typically decision trees, are built sequentially. Each model attempts to correct the errors made by the previous models, and this process is iterated. The final strong model is an ensemble that combines the outputs of all the individual weak models to predict the target (label) variable based on the given input features [8].

Our model is actually split into three different models - one for serves, one for returns, and one for the rest of the rally. Each of these instances in the point have different contexts so it seemed appropriate to break them apart. For every shot entry in each dataset, we will seek to predict an expected reward, or "**xR**", based on the environment and action (level of aggressiveness). We will then create a dataset of every possible combination of environment and actions (the dataset would have 92400 entries as mentioned above). These values will be fed into the model which will output a corresponding **xR**. For every environment, we will find the action that yields the highest **xR**. This action will be our recommended level of aggression for players in that situation. All of the models were trained on data from 2020-2022 and tested on data from 2023. The following sections detail the process for creating the models.

5.1 xR - Serves

Both the score and ranking variables are considered *factor* variables. In other words, they are categories rather than continuous numbers. We converted every bin at the point, game, and set level into a dummy variable. This meant each bin had its own column with a binary output. This transformation made our environment go from five factors to 27 binary variables.

After running our model and tuning its features, we settled on the following parameters:

- **Number of decision trees** - 250
- **Learning objective** - Regression with squared loss
- **Early stopping rounds** - three (If the model doesn't improve after three rounds then it is stopped)
- **Maximum tree depth** - six
- **Learning rate** - 0.18

Our model had a root mean-squared error of 4.12 indicating some amount of variation between the expected and actual reward. This is actually positive, since we want to emphasize which level of aggressiveness is best in a given situation. If the actual reward of a shot is a lot higher than the expected reward, for example, this indicates that the player hitting the shot may have gotten lucky and in the long-run, that level of aggressiveness in that situation might be sub-optimal. **Figure 1** visualizes the top eight variables in our model. No surprise, aggressiveness is the most important. It's also a good sign that hitter and opposition ranking bin one are among the top. We hypothesized that being a top player or playing against a top player would influence someone's aggressiveness.

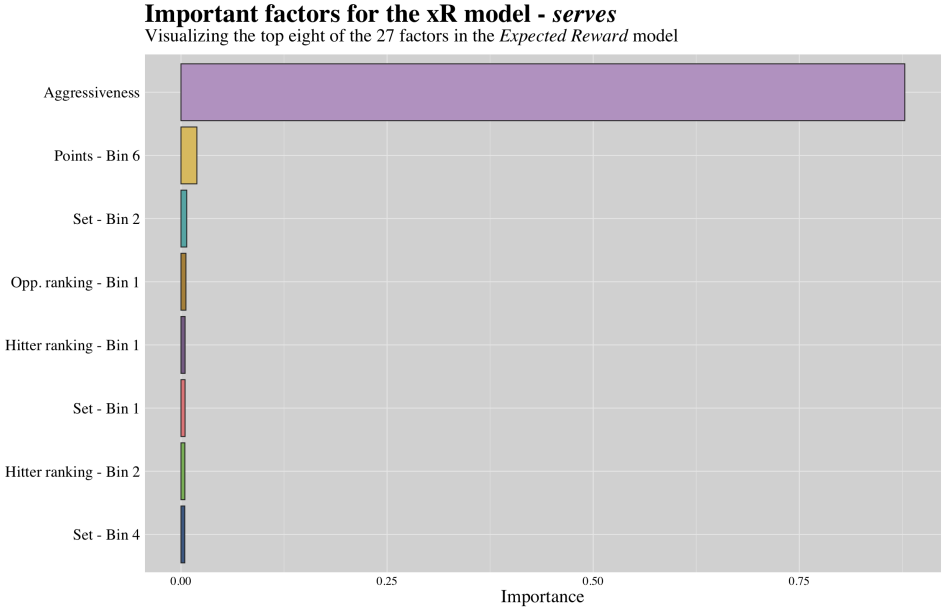

Figure 1: Eight most influential factors in the *serve* xR model

5.2 xR - Returns

Next, we will conduct the model for the return shot. We kept the same parameters as the previous model. For this model, we observed a root mean-squared error of 4.77. **Figure 2** visualizes the most important factors in the returns model. An interesting observation is that opposing rankings appeared multiple times, but hitter ranking did not appear at all. This makes sense since returns are literally the response to the opposing player's serve. If the opposing player was ranked higher, it would make logical sense that the returning player would change their approach. It is also worth noting that these variables had a larger influence on the outcome than the previous model.

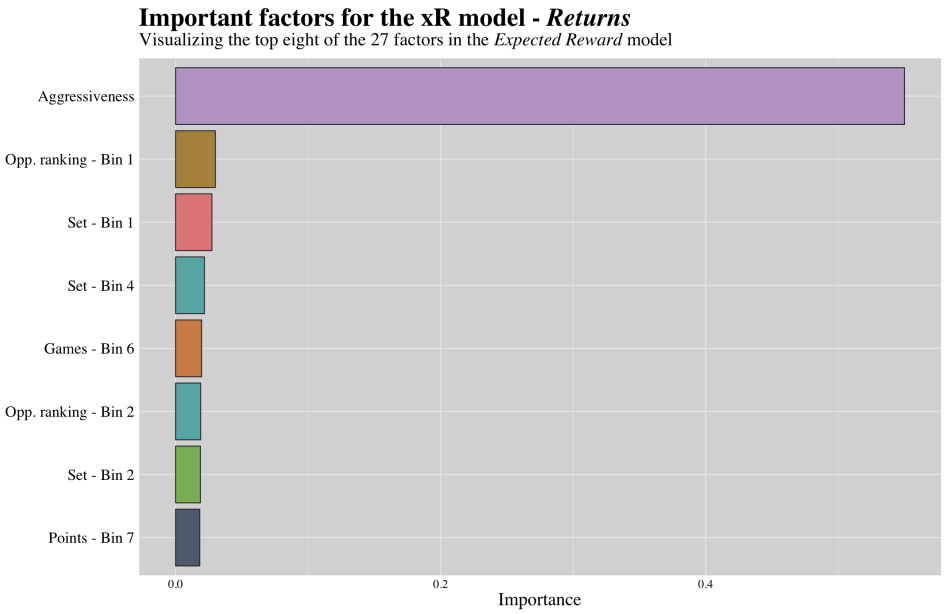

Figure 2: Eight most influential factors in the *return* xR model

5.3 xR - Rallies

Our final model is the XGBoost in the context of rallies. This includes every shot in the point except for the serve and the serve's return. Once again, we kept the same parameters as before. This model had a root mean-squared error of 5.07. **Figure 3** visualizes the most influential factors in the model. No huge surprises here, but once again, we see that being a top-ranked player and facing a top-ranked players are huge indicators of aggressiveness.

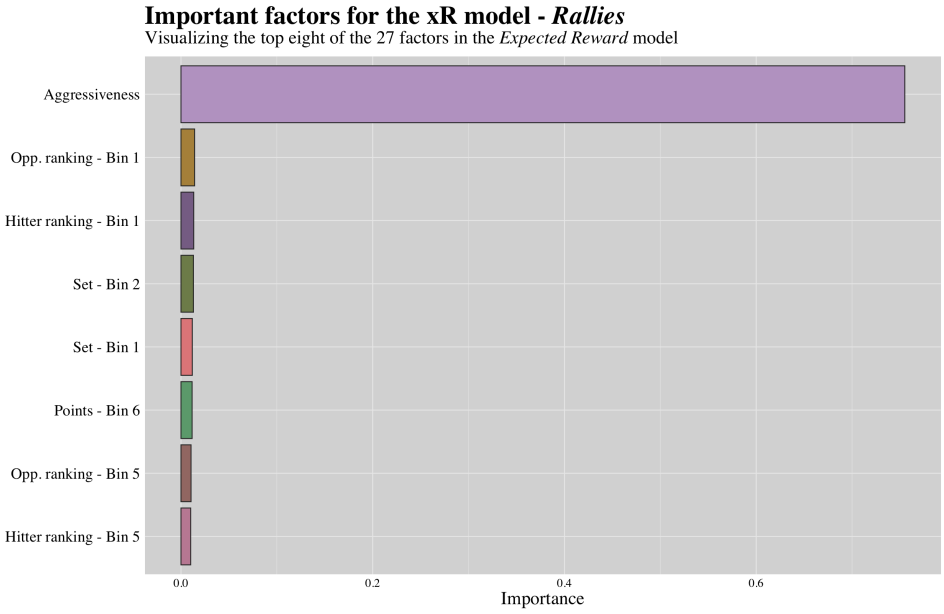

Figure 3: Eight most influential factors in the *rally* xR model

5.4 Determining the optimal aggressiveness

Now the real test is to determine the optimal aggressiveness for every situation. As mentioned earlier, there are **4200** different environments that a player can find themselves in and 46200 different environment-action pairs that a player can take. We created a dataset of all 46200 pairs and fed it into our three XGBoost models. The results represent the expected reward for every action in every situation. Now to determine the optimal aggressiveness, we found the aggressiveness level that yielded the highest expected reward in each environment.

If an environment that had multiple aggressiveness levels with equally optimal rewards, we took the lower aggressiveness level since lower aggression is associated with lower risk. **Figure 4** displays the frequency of every aggressiveness level. The first thing that pops out is that high levels of aggressiveness are optimal in an overwhelming majority of instances.

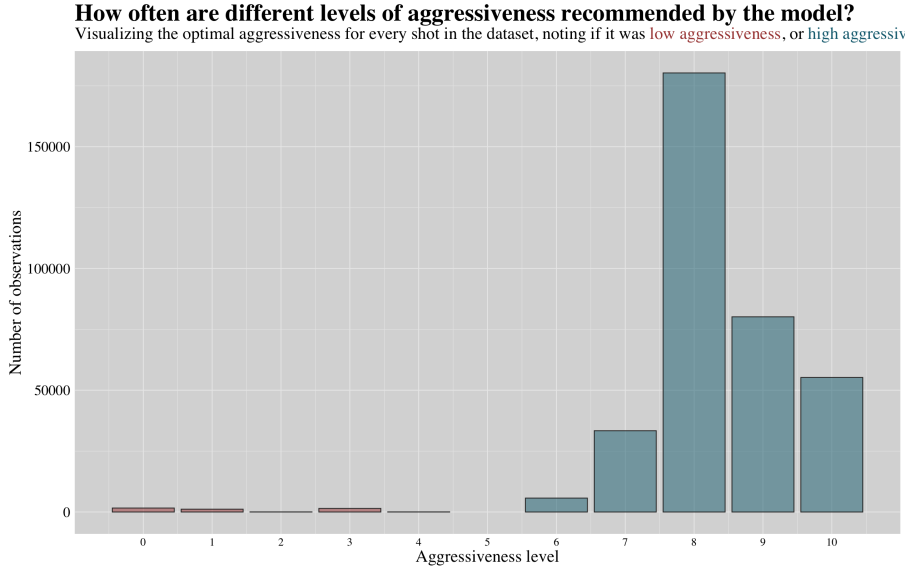

Figure 4: Frequency of each aggressiveness level

Next, we wanted to understand the reward distribution at each aggressiveness level. To properly visualize this, we split up our environment into score differential and rank differential.

Beginning with score differential, we had the point differential, game differential, and set differential at any given shot. We decided to create a **weighted sum model** to find an overall score difference in the match. Since it normally takes four points to win a game and six games to win a set, let

$$\Delta_{ovr} = \Delta_{point} + 4 \times \Delta_{game} + 24 \times \Delta_{set}$$

Where Δ_{ovr} is the overall difference, Δ_{point} is the standardized point difference for the current game, Δ_{game} is the game difference for the current set, and Δ_{set} is the set difference for the match. For example, if someone was winning two sets to none, the start of the third set would mark a differential of 48. **Figure 5** visualizes the reward distribution of the observed optimal aggressiveness at every score differential. Note that there could be multiple aggressiveness levels at any given differential.

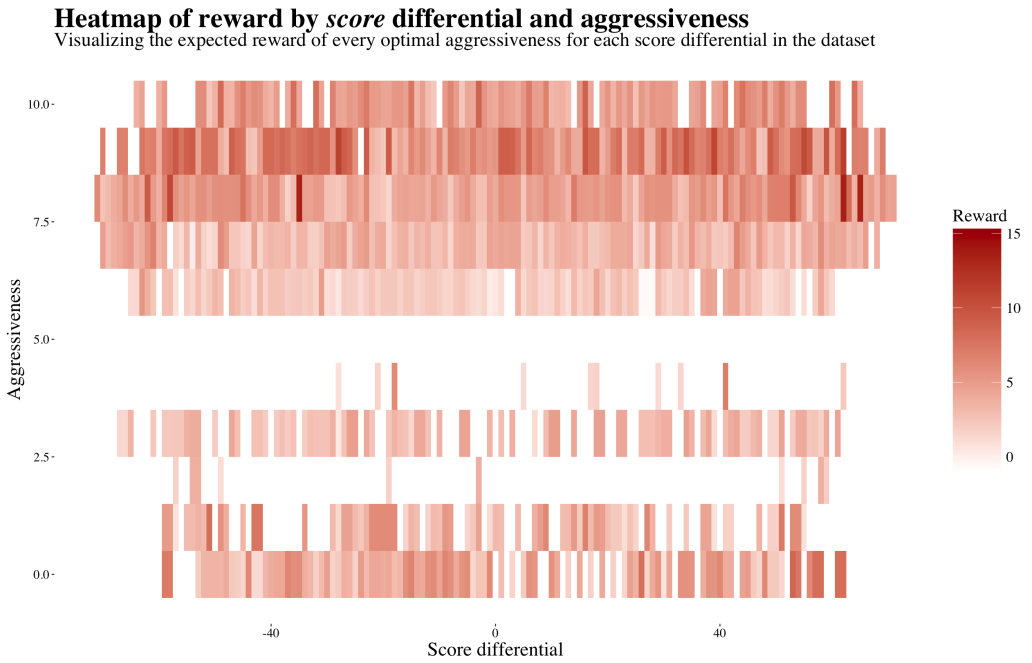

Figure 5: Visualizing the optimal aggressiveness for every score difference

This looks somewhat complicated, but there are two main observations we can take away. First, the model recommends higher levels of aggressiveness more often than not as seen by the frequency discrepancy. Second, the higher levels of aggressiveness are associated with **larger rewards**. We can also see that if a player is winning by a lot, the model associates super low levels of aggressiveness with a larger reward. The opposite is true if a player is losing by a lot.

Next, we examined the reward distributions for the different aggressiveness levels based on rank difference. Recall that we separated player and opposition rank into five different bins when creating our models. We subtracted player rank from opposition rank to create the **rank difference** variable. If a player has a rank difference of four, for example, that indicates that they are a top ten player playing against an unranked player. **Figure 6** visualizes the reward distribution of every aggressiveness level observed in a given rank differential. Similarly with match situation, we will see multiple aggressiveness levels for each rank differential.

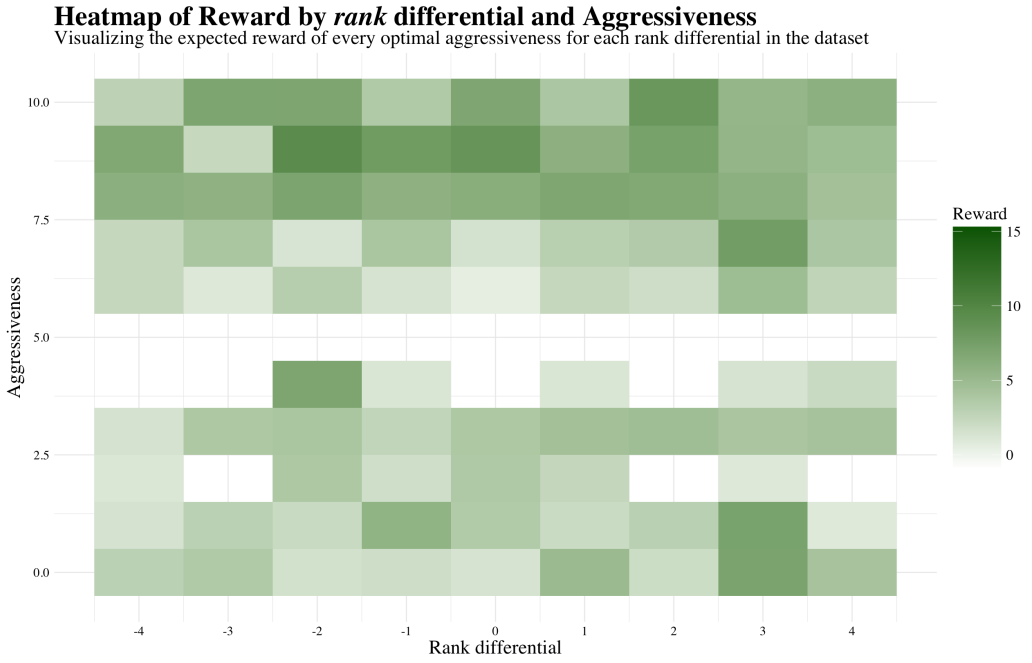

Figure 6: Visualizing the optimal aggressiveness for every rank difference

This heat map is very similar to the previous. In addition to the main takeaways that were mentioned earlier, we can see that in some instances, if players are a lot better than their opponent, the model recommends they should play with lower levels of aggressiveness. The opposite is true for players whose opponents are much better than themselves. Perhaps playing with high levels of aggressiveness could be grounds for a potential upset.

6 Discussion

6.1 Limitations

While we were satisfied with our results, there are a couple limitations to our methodology. First, our model may have been naturally biased towards associating higher levels of aggressiveness with a higher reward. Certain shots, like aces for example, are associated with the highest level of aggression and are guaranteed to have the highest reward. This is not necessarily wrong, but it means that the model is trained on the premise that aggressiveness yields higher rewards in a few instances.

Secondly, we had to generalize when categorizing shots as aggressive. There could be certain shots we categorized as aggressive that, in other contexts, could be argued as conservative. This is because we did not have access to potentially important data, such as player location and ball speed. Given this information we could have created a more accurate aggressiveness metric.

6.2 Future applications

As mentioned earlier, the implementation of both player and ball tracking data could provide a more accurate estimate of a shot's aggressiveness. We can also apply our research to the Women's Tennis Association. The ultimate goal is to turn our research into an app where a player or coach would input the in-game situation and receive a recommendation on the optimal level of aggressiveness and what shots they could play.

7 Conclusion

Our research has revealed a significant correlation between aggressiveness and success in tennis. Despite this, players may choose not to be aggressive in certain situations for several strategic reasons. First, aggressiveness carries inherent risks, such as increased chances of unforced errors, especially in high-pressure moments where precision is crucial. In tight matches or when facing a break point, players might prioritize consistency and control over aggressiveness to avoid giving away free points. Additionally, opponents may be adept at countering aggressive play, particularly if they are strong defensive players who thrive on their ability to turn the opponent's power against them. In such cases, a player might opt for a more measured approach, aiming to outlast the opponent in longer rallies or force them into uncomfortable positions rather than going for outright winners. Furthermore, the physical and mental demands of aggressive play can be taxing, and players may conserve energy during longer matches or in conditions where fatigue could become a significant factor. Ultimately, while aggressiveness can lead to success, players must weigh the benefits against the risks and adapt their strategy to the specific context of the match, their opponent, and their own physical and mental state. While players shouldn't completely change their playstyle based on our research, tennis is a game of momentum and tenacity, where even the smallest advantage can be the difference between a win and a loss. We believe that our model has the potential to provide that competitive edge.

References

[1] Kevin Mitchell, "Roger Federer v Rafael Nadal: Wimbledon's greatest final a decade on", *The Guardian*, July 1, 2018. Available: https://www.theguardian.com/sport/2018/jul/01/roger-federer-rafael-nadal-wimbledons-greatest-final-a-decade-on.

[2] Jeff Sackmann, "The Match Charting Project: Quick Start Guide," *Tennis Abstract*, September 23, 2015. Available: https://www.tennisabstract.com/blog/2015/09/23/the-match-charting-project-quick-start-guide/.

[3] J. McQueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Computer and Chemistry*, vol. 4, pp. 257-272, 1967.

[4] Geeks for Geeks, 'Elbow Method for optimal value of k in KMeans," *Geeks for Geeks*, May 10, 2023. Available: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/.

[5] Ajitesh Kumar, 'KMeans Silhouette Score Python Example," *Analytics Yogi*, April 26, 2023. Available: https://vitalflux.com/kmeans-silhouette-score-explained-with-python-example/.

[6] Guardian Sport, "Final Sets in All Four Grand Slams to Be Decided by 10-Point Tie-Break," *The Guardian*, March 16, 2022. Available: https://www.theguardian.com/sport/2022/mar/16/final-sets-in-all-four-grand-slams-to-be-decided-by-10-point-tie-break-tennis.

[7] ESPN, "Men's Tennis ATP Rankings 2024," *The Guardian*, 2024. Available: https://www.espn.com/tennis/rankings.

[8] NVIDIA, "What is XGBoost?," *NVIDIA Glossary*, Available: https://www.nvidia.com/en-us/glossary/xgboost/.