Jason Haley

Gabriel Hicks

10 December 2024

DTSC 201

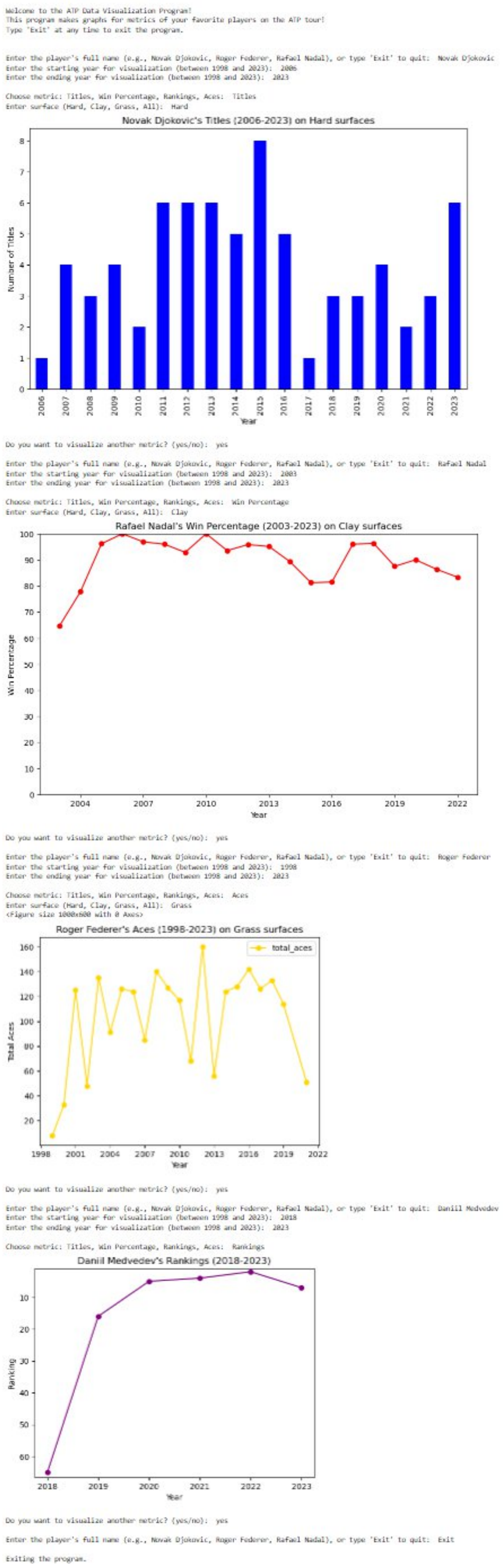
Project Report

I. Project Overview and Implementation

Our project was about using a comprehensive ATP (Association of Tennis Professionals) dataset to answer our research question of "Who is the GOAT of Professional Tennis?" As there is a lot of discussion in the tennis community of the big 3 – a combination of the three greatest players in the world: Novak Djokovic, Roger Federer, and Rafael Nadal – we wanted to use statistical analyses to try to come to a definitive conclusion to this question.

For our implementation to answer this question, we decided to make an interactive data visualization program with the dataset. In this program, the user could enter information specific to the metric that they wanted to visualize. The information that the user entered for this was the following: player name, starting year for visualization, ending year, the metric, and the surface that they wanted the statistical visualizations for. The user then was given a graph according to the information that they typed in. This implementation involved two separate programs: Jason's program involved visualizing metrics for a single player while Gabriel's program gave the option for metrics that involved multiple players.

In Jason’s program (Figure 1), the user could type in the name of any player in the ATP (not just the members of the Big 3). It then would ask the user to input the starting and ending years any time from 1998 to 2023. metrics that were able to be displayed in this program were the number of titles per year, the win percentage per year, rankings (measured at the beginning of every year), and aces. The final question the program would ask is the surface that the user would like the visualization to be for. They could choose between hard courts, clay, grass, or all surfaces. A graph is then displayed assuming all the information was entered in correctly for the visualization the user requested. The user is then asked if they would like another metric to be visualized, and can enter “yes” or “no” based on their preferences. A couple of additions were added to the program to make it more user friendly. One thing is that the user may type “Exit” at any time if they wish to exit the program. Also, the program is not case sensitive, so if the user types in a name with incorrect capitalization, the program will still find the correct name. In addition, the charts (besides the aces and rankings metric) update the color according to the surface you choose (hard court is blue, clay is red, grass is green, and all surfaces is blue). There are also numerous try and exception pairs that will output an error message if the user inputs any information incorrectly such as a name that is not in the database, a year outside the program’s bounds, a metric not included, upon many others.



```
Welcome to the ATP Data Visualization Program!
This program makes graphs for metrics of your favorite players on the ATP tour!
Type 'Exit' at any time to exit the program.


Enter the player's full name (e.g., Novak Djokovic, Roger Federer, Rafael Nadal), or type 'Exit' to quit:  Novak Djokovic
Enter the starting year for visualization (between 1998 and 2023):  2006
Enter the ending year for visualization (between 1998 and 2023):  2023

Choose metric: Titles, Win Percentage, Rankings, Aces:  Titles
Enter surface (Hard, Clay, Grass, All):  Hard
```

```
Do you want to visualize another metric? (yes/no):  yes

Enter the player's full name (e.g., Novak Djokovic, Roger Federer, Rafael Nadal), or type 'Exit' to quit:  Rafael Nadal
Enter the starting year for visualization (between 1998 and 2023):  2003
Enter the ending year for visualization (between 1998 and 2023):  2023

Choose metric: Titles, Win Percentage, Rankings, Aces:  Win Percentage
Enter surface (Hard, Clay, Grass, All):  Clay
```

```
Do you want to visualize another metric? (yes/no):  yes

Enter the player's full name (e.g., Novak Djokovic, Roger Federer, Rafael Nadal), or type 'Exit' to quit:  Roger Federer
Enter the starting year for visualization (between 1998 and 2023):  1998
Enter the ending year for visualization (between 1998 and 2023):  2023

Choose metric: Titles, Win Percentage, Rankings, Aces:  Aces
Enter surface (Hard, Clay, Grass, All):  Grass
<Figure size 1000x600 with 0 Axes>
```

```
Do you want to visualize another metric? (yes/no):  yes

Enter the player's full name (e.g., Novak Djokovic, Roger Federer, Rafael Nadal), or type 'Exit' to quit:  Daniil Medvedev
Enter the starting year for visualization (between 1998 and 2023):  2018
Enter the ending year for visualization (between 1998 and 2023):  2023

Choose metric: Titles, Win Percentage, Rankings, Aces:  Rankings
```

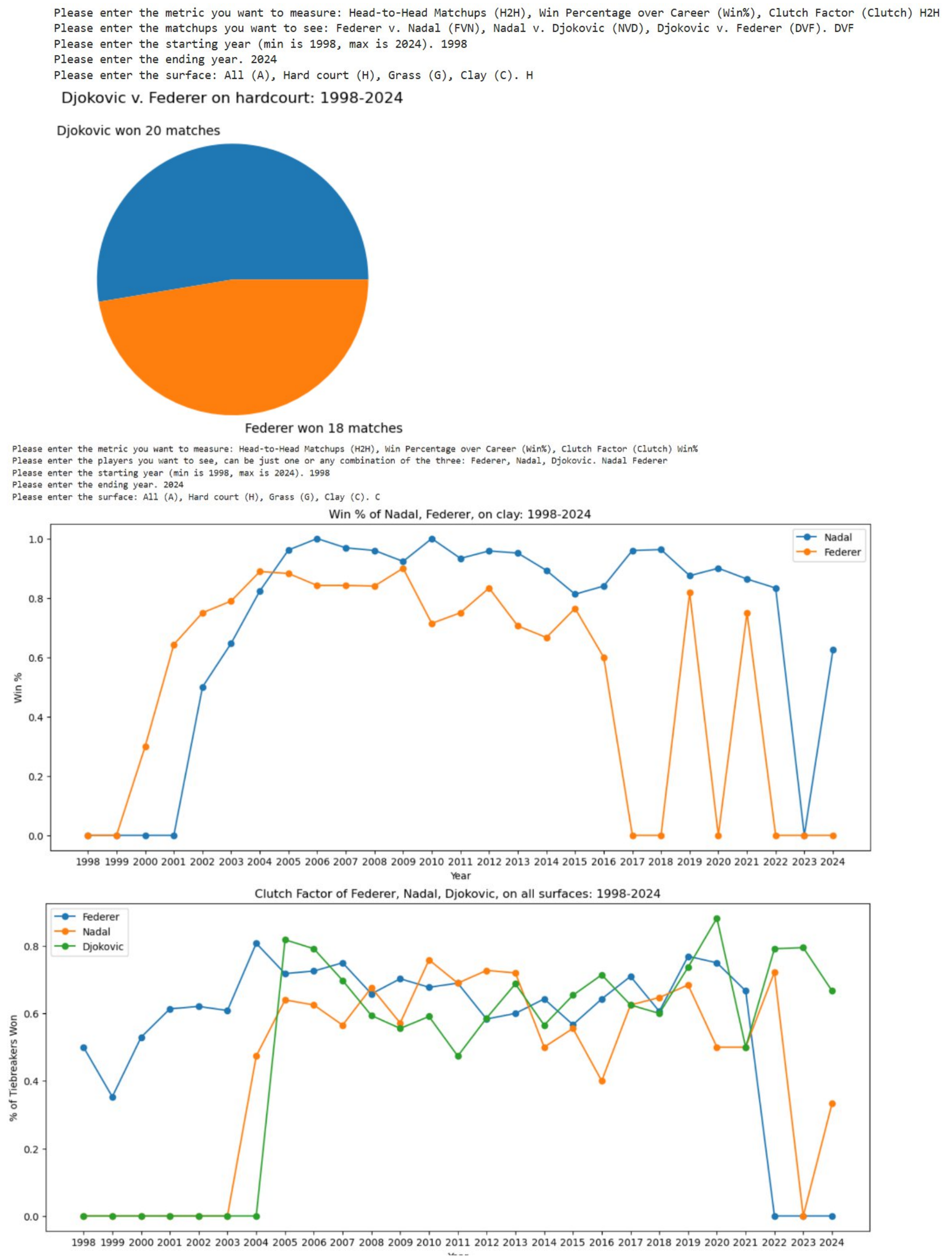
```
Do you want to visualize another metric? (yes/no):  yes

Enter the player's full name (e.g., Novak Djokovic, Roger Federer, Rafael Nadal), or type 'Exit' to quit:  Exit

Exiting the program.
```

Figure 1. Example Implementation of Jason's program using ATP Dataset (Kaggle, 2024)

In Gabriel’s program (Figures 2-4), the user could type in the name of any player in the Big 3 specifically to get visualizations of these players. However, this program was able to take multiple player’s names as inputs and put their data onto the same graph to allow for a direct comparison of the Big 3. This helps answer the research question since players are being compared directly side by side. Gabriel’s program is very similar to Jason’s as you can enter similar information. First the program asks for the metric you want to be visualized, then the names of the players you want the visualization done for, the starting and ending year, and then the surface you want the metric to be for. It then outputs a graph according to the information that was typed in. If any information is entered incorrectly, the program enters automatically outputs the message, “Make sure to only enter the last names of players, single years, or the characters in parentheses based on what you want to see.”



```
Please enter the metric you want to measure: Head-to-Head Matchups (H2H), Win Percentage over Career (Win%), Clutch Factor (Clutch) H2H
Please enter the matchups you want to see: Federer v. Nadal (FVN), Nadal v. Djokovic (NVD), Djokovic v. Federer (DVF). DVF
Please enter the starting year (min is 1998, max is 2024). 1998
Please enter the ending year. 2024
Please enter the surface: All (A), Hard court (H), Grass (G), Clay (C). H
```

```
Please enter the metric you want to measure: Head-to-Head Matchups (H2H), Win Percentage over Career (Win%), Clutch Factor (Clutch) Win%
Please enter the players you want to see, can be just one or any combination of the three: Federer, Nadal, Djokovic. Nadal Federer
Please enter the starting year (min is 1998, max is 2024). 1998
Please enter the ending year. 2024
Please enter the surface: All (A), Hard court (H), Grass (G), Clay (C). C
```

Figure 2-4. Example Implementations of Gabriel's program using ATP Dataset (Kaggle, 2024).

II. Project Goals, Data source/Dataset, and Toolkits

Our data we used for this project was sourced from Kaggle.com. The dataset was named "Huge Tennis Database" and was compiled together by Guillem SD. However, he gave credits to the original data set creator Jeff Sackman. The toolkits and libraries that we used to implement our program were pandas, matplotlib.pyplot, matplotlib.ticker, and numpy. The libraries pandas and numpy were mainly used for data wrangling and analysis. On the other hand, the matplotlib toolkits were implemented to create our visualizations and customize them.

The dataset was created just before the ATP started. The ATP started in 1972, but this dataset dates back to 1968. It has all the matches up until May 2024. The match files in our dataset had the most data and had information such as tournament level, winning player, surface type, and many other key information used for our program. The rankings files simply showed players' rankings updated weekly. The players files give descriptions of players and different attributes about them such as their country.

III. <u>Issues and Solutions</u>

One issue that came up was how we would implement a GUI or dashboard into our project. We ultimately decided against having one, but in our initial plan we did plan to have it implemented a few weeks before our final presentation. The issue was that we had never coded a GUI before, so we didn't know how to create one or how to actually implement one into our project in a way that wasn't clunky or unnecessary. One problem that I (Gabriel Hicks) personally had was that I wasn't sure how I would display plots or images through the GUI and not through the program itself. In my mind I was imagining just a bunch of buttons that led the user to different graphs depending on the metric and player and whatever else they wanted to see, and I worried that would result in too many buttons for the user to click, making the GUI either clunky or tedious. We both watched several videos on how to make GUI's to try and understand how we could use one in our program, but we were both unsure of how a GUI would work in our program or if there was a simpler solution. The way we resolved this issue was to ultimately scrap the idea of implementing a GUI or a dashboard and instead sticking with Dr. Huang's suggestion of having the user directly interact with the program, which we decided to do with text prompts for the user to fill out. This really increased the scope of the project and made it more substantial, and in hindsight a GUI may not have been needed for this particular project, or at the very least the project didn't need a GUI that was hastily added into the program a week or two before the final presentation by two people who had never made one before.

Another issue we had was implementing the user interactive portion to our project. At first we only made one or two plots for each metric we worked on, and so the parts of our code that were responsible for creating the plots really only needed to be run once. However, now that we needed to create a new graph for each new entry of user information, we needed to figure out how to transform our parts of code to have them create different graphs based on user input instead of just based on what we had originally designed them to do. The way that I (Gabriel) personally tackled this new problem was to change how I read in the csv files of match data. At first I read in the files from 1998-2024 and then combined them into a single dataframe so that the functions I created for collecting variables for each metric would only need to be run once. However, now that users would need to enter a year range, I had to consider each year of data separately, and so I removed the code that combined all of the dataframes into one. Then, I modified those functions with a parameter for the year, and then called those functions for each dataframe in a list of the dataframes I had created previously. I then had to store a whole lot of new variables to access for analysis based on user input, so I created a hierarchy of metric, surface, year, and player for my dictionaries that stored data. Each metric I worked on had their own sets of dictionaries, each surface had a dictionary of its own, each year from 1998-2024 was a key in each of these dictionaries, with their value being another dictionary with the players as keys and finally the variables as values. I needed a consistent system like this to keep track of where the data was actually being stored in case I needed to change something, and I kept this same hierarchy for each metric I added so that if I needed to fix something with one metric, I would do roughly the same thing for the other metrics instead of going through the weeds of inconsistent storing methods and categorization.

IV. <u>What we have learned</u>

From this project I learned how to work on a long-term coding project. I (Gabriel) had never worked on a project like this before, so I learned a decent amount about time management and enforcing deadlines so that I do a decent amount of work each week instead of trying to cram it all into a few weeks. I also gained a new appreciation for coding from this project, as before I had only really worked on homework problems and exercises while coding. Now that I have completed an actual project that serves a purpose beyond being something for a professor to grade, I realized how fun and satisfying programming can be when you truly care about what you are doing. I always had a good time coding in the past, but being able to say that I really created something from scratch is a really satisfying feeling.

V. Real World Application

Our project would be extremely useful in the real world as it takes data from all ATP tennis matches and analyzes the information to create helpful visualizations. Besides being used to settle the GOAT debate, one of the ways the project can be applied in the real world is by coaches and players looking at various statistics of either themselves or opponents. For example, if a certain player feels like they are not hitting their serve as well as they used to, they could come use this program and see if their ace rate has actually gone down in recent years. They can use this to gauge the types of aspects of their game they need to focus on, whether it be their serve or a particular surface they do not perform well on. This could easily be expanded to include other statistics if given access to those types of data. It could be used for all types of strokes (forehand, backhand etc.), matchups with all players, and even the time that a player has spent on court. These types of metrics are vital to professional players as getting a slight edge over the rest of the playing field can be the difference between winning and losing a tournament.

VI. How this Experience can translate to Workplace or Future Projects

One way that this project could be used in the workplace is as a general template to read in various csv files a worker may need to access and work with, and to generate quick plots of the data in those files. Being able to create entirely new graphs quickly based on user input could be immensely useful for presenting your findings during a meeting, and being able to answer simple questions regarding the data you're working with by answering a few prompts would save time while offering legitimate answers. The user would have to alter which specific cells the program is accessing, among other things like editing the titles of graphs and the names of variables, but the general structure would stay the same. Potentially the user could make it so that they enter the csv file or set of csv files they want to be analyzed along with the specific cells they want accessed so that all they have to do is upload the files and enter in those prompts to receive plots instead of altering the code itself for each new csv file they work with.

VII. **Source List**

Huge Tennis Database (kaggle.com)

FAQ about PIF ATP Rankings | ATP Tour | Tennis

ATP Official Rulebook | ATP Tour | Tennis (Section IX)

Official Site of Men's Professional Tennis | ATP Tour | Tennis

How Do Tennis Rankings Work? (Easy Guide) - My Tennis HQ

https://www.itftennis.com/media/11556/itf-points-tables-2024.pdf

https://en.wikipedia.org/wiki/Big_Three_(tennis)