Data Visualization

Shiny

Authors:
Julio Martínez Bastida
Carlos Sánchez Velázquez
Rafa Sojo García

1 Introduction

In the following pages we will try to answer different questions by means of data visualisations. In our case, we are going to use a tennis dataset that can be obtained from CITE, in which we can find both information relevant to the tournaments played between 2010 and 2019, as well as most of the matches played in those tournaments. In addition, the views presented below follow the theoretical developments explained in the course lectures, so we will try to explain our design decisions as far as possible.

2 Problem characterization in the application domain

Professional tennis is currently ruled by the Association of Tennis Professionals (ATP). This association establishes a series of different official tourneys that are organised every year and distribute an amount of points to the players that allow them to climb positions in the ATP official ranking. There are 4 official categories of tourneys according to the amount of points they distribute, which, in increasing order of importance are: ATP 250, ATP 500, Masters 1000, and Grand Slams. At the end of the year it is common to analyse which players have won the different tournaments, and how they have been distributed among the rest. In relation to that it is pretty frequent to see whether the titles have been monopolised by a little amount of players (for example, if the 4 Grand Slams have been won only by 1 or 2 players), or otherwise, most of the titles have been won by different players, which is in fact, analysing the part-to-whole relationship of the winners with the total amount of tourneys. Apart from that, it is also common to analyse their performance considering different statistics such as the number of aces or the total points won, something that could vary considerably from player to player depending on many other factors like for instance the surface of the court, the round of the match or the type of tournament. This lead us to three main questions:

- Q1 : What is the part-to-whole relationship of the winners of all tourneys of a certain category? What is its evolution over the years?
- Q2: How have the players evolved over the years? Does the surface affects in that matter? Is there a noticeable difference if the players end up losing the match? What if they end up winning?
- Q3: Is there any hidden pattern for a certain stat given its corresponding round and tournament? Does the surface affects in that matter?

For the first of these three, we plan to make a visualisation that could help us to gain insight in determining years or periods in which there has been more competence among the players or years in which a small group have dominated the tennis circuit, comparing also whether that occurred for all categories of tourneys or not. However, with the second visualisation we are more interested in comparing their evolution in a bunch of different scenarios, for example, checking how the three great tennis players of the moment, Roger Federer, Novak Djokovic and Rafael Nadal, have developed their skills under different surfaces, victories, or defeats. On the other hand, with the third question we want to focus the attention on, perhaps, something more abstract at first, what can be the main differences over the stats considering the round and tourney of that match. With the first two questions we are gaining insight with respect to the circuit and the players, while for the third more about how the statistics are affected by the match, whose patterns could implicitly hide information about the level of the athletes within the different tourneys

3 Data abstraction

For this task we have used some arranged tabular data divided in two table datasets. The first, contains 668 rows (items), each one representing a tennis tournament that took place in a certain year from the period of 2010 until 2019, and a total of 30 different attributes of all different types, quantitative, ordinal

and categorical. In contrast, the remaining dataset contains relevant information about the statistics achieved by each player at each match, resulting in 42447 items and 125 attributes of the same type as before, both containing their respective identification attributes for matching them. The most relevant attributes for our visualisation are the following:

- *tourney_type*: Indicates the category of the tournament. This might seem like a categorical attribute, however, it contains an implicit order, therefore, an ordinal attribute.
- *tourney_round_name*: Indicates the round of the match in that tournament. Same situation as before.
- *tourney_surface*: Indicates the tournament surface (categorical).
- *tourney_name*: Indicates the name of the tournament (categorical attribute).
- *singles_winner_player_name*: Indicates the name of the player that won that tournament (categorical attribute).
- *year*: The year of the tournament, since this is an attribute related with time, it could be considered a quantitative or an ordinal attribute, with a sequential ordering direction, for this task and visualisation we will consider it as ordinal, since we only have information of 10 years.
- *winner_name* and *loser_name*: Indicates the name of the player that won the match and lost the match, respectively (categorical attribute).
- *winner_pctg_STAT* and *loser_pctg_STAT*: Where STAT is a substring that depends on the chosen statistic and can take 13 different values.The first of the attributes refers to the value of the statistic for the winner of the match while the second for the loser (quantitative attribute).
- *match_duration*: Indicates the duration of the match (quantitative attribute)

To solve the first task, only three of the above mentioned are necessary, *tourney_type*, *singles_winner_player_name* and *year*, while for the second visualisation, *tourney_name*, *tourney_surface*, *year*, *winner_name* and *loser_name*, *winner_pctg_STAT*, *loser_pctg_STAT* and *match_duration*, in addition to the *tourney_type*. Likewise, for the third visualisation all previous attributes are needed, with the exception of *year*, *winner_name* and *loser_name*. *tourney_round_name* is also needed.

All the attributes of the first task are from the first dataset, however, for the remaining two we have had to join both using the *tourney_name* as identifier for the *tourney_type* and *tourney_surface* attributes, as they are not included in the one containing data about the match statistics.

Finally, to solve the different tasks we have had to perform an aggregation over the corresponding dataframe for each case. With respect to the first visualisation, to sum up all tournaments won by each player each year, therefore, generating a new quantitative attribute, while for the second for obtaining the average statistic grouped by year and by player. Similarly, the third one uses the same average, but grouped by tourney type and round name.

4 Task abstraction

Once the main problem has been exposed, we should carry out a task abstraction in order to justify the use of a visualisation tool to solve these problems (why should we use a visualisation?).

4.1 Question 1

It is true that with this visualisation we will present information which is already known, such as the winners of tournaments, however, the main objective is to discover or extract new knowledge that was not explicitly in the raw file, such as the evolution of the mentioned part-to-whole relationship, and discovering in which years there has been more competence in the tennis circuit, so the main task would be to discover new knowledge. This new knowledge could generate us a new hypothesis about the data (i.e., maybe these years the Grand Slams were all won by one player because...) or to confirm a previous one. In this task we know that we are looking for the winners of known tournaments in known years, however, at first we do not know which are these players, so the class of search of this task is a browse search, since we already know where to find the information (we know the tournaments and the years) but we do not know what exactly we are looking for (we do not know the winners of those tournaments. Afterwards the defined target (winners of all tourneys of a certain category for every year) has been found, a query is performed over them, in this case we want to query all players, to find their part-to-whole titles relationship, so the type of query we are performing is a summarising query, since the scope is all possible targets. More specifically, with this task our target is to get the percentual distribution of the winners of the tournaments, and furthermore, the evolution of this over the years.

4.2 Question 2

Unlike the previous visualisation, for this one the information shown must be obtained as the average of the data for each match. However, the knowledge we are interested in is the comparison of the graphs between players, which allows us to corroborate known facts, such as that Rafael Nadal tends to play longer matches than other players, or generate new hypotheses such as that he is a player who takes fewer risks when serving. For the development of this application, the information is known at the time of searching, both the players and the statistics, as well as the particular context desired, i.e. the surface and the final result of the match. As we need to search for all the matches of the selected players in the context the user requires, this is a Browse search At last, the main target of this visualisation is to check the evolution of the different players over the years in the contexts specified above, using the average of one attribute per match. This leads to the fact that, in this case, the query is a comparison query.

4.3 Question 3

Likewise, the last question aims to use the statistics averages of each match, but grouped together by tourney and round instead of being obtained for each player, therefore, using the resulting visualisation to also discover new knowledge like for instance whether the number of double faults increases or decreases by round, and whether the court's surface affects on that aspect or not. Thus, verifying previous hypotheses such as the fact that the surface has an impact on the overall game, and generating new ones about the impact of the round in the stat. With respect to what and where we are looking for, we know exactly that we want to look at the variation in the different statistics under the mentioned conditions, but we do not know what is going to look like the overall summary, or whether we will find clusters within that visualisation. Therefore, we are talking about a Browse search in that aspect. On the other hand, here, the query refers to a single target since it requires identifying the characteristics of each individual statistic for the whole round and tourney matrix. Even though we could later check some others, it would always be one at a time.

5 Visual encoding and interaction

5.1 Question 1

As mentioned above we want to represent either a part-to-whole relationship and its evolution over the years. To represent a part-to-whole relationship, we know we can use a normalised stacked barchart or a

pie chart, and if we wanted also to visualise its evolution, it would be necessary to plot several of them aligned, since "eyes beat memory", it is better to have the option to see all at the same time to compare. We decided to use several normalised stacked bar charts, the first reason is that the stacked barchart uses length as visual channel to represent the quantitative attribute, and the pie chart uses angle for that task, which is a less effective channel to the human eye, the second reason was that pie charts require way more space than stacked barcharts. Another interesting option was to use a streamgraph, however, we only had 10 years, which was not enough.

As mentioned, several normalised stacked barcharts were used, this type of idiom uses as marker a glyph (bar) formed by several area submarkers, these submarkers encode the different "parts" of the "whole" (in this case, the percentage of tourneys won) with its length channel, and encode a player with a colour chanel, the length of the glyph also represents the percentage of titles won which will be 1 for all bars, since it is a normalised barchart. Finally, the years are encoded with the spatial position of the bars.

We want to compare for the different years, so the most important attribute to encode is the year, in this case, with the most effective visual channel to encode ordinal data which is spatial position, after that the second most effective channels for percentage of won tournaments and player are length and colour respectively.

It is also important to mention that the maximum recommended number of categories to encode with colour is 12, but we decided to use only 10 because 12 were also quite confusing in our opinion. However, in several situations, our visualisation contains more than 10 players, in order to solve that, we decided to create a category for that attribute called "Others" which would include all the players, except the first 9 players with the most titles won.

To make it possible to know the information encoded in the category "Others" for a selected year, it was decided to use a second visualisation just under the first one. This visualisation needs to encode the percentage of titles won by all players in the category of "Others". To do so, a bar chart was selected, which encoded the players (categorical attribute) with the spatial position of the bars (most effective channel for the most important feature), and the percentage of titles won was encoded with the length of the bars.

With this visualisation we can observe the evolution of our target for a certain category of tournament. Having the same evolution for all the different types of tournaments at the same time would require too much space, so it was decided to have the possibility to change the visualisation by means of interaction. More specifically, a select option was included to change the type of tournament so that when it changes, the data to show is filtered according to that category.

That is not the only interaction our visualisation offers. As mentioned, part of the information is aggregated when the visualisation needs a high number of colours to encode all the players using the category of "Others", the user has the option to select one year and in a certain way "disaggregate" that information in a second chart that indicates the content of that category, this interaction is done with a select item that includes all possible years.

Finally, both plots are juxtaposed, being the normalised stacked barchart over the normal barchart. The resulting idiom can be appreciated in the figure fig. 1

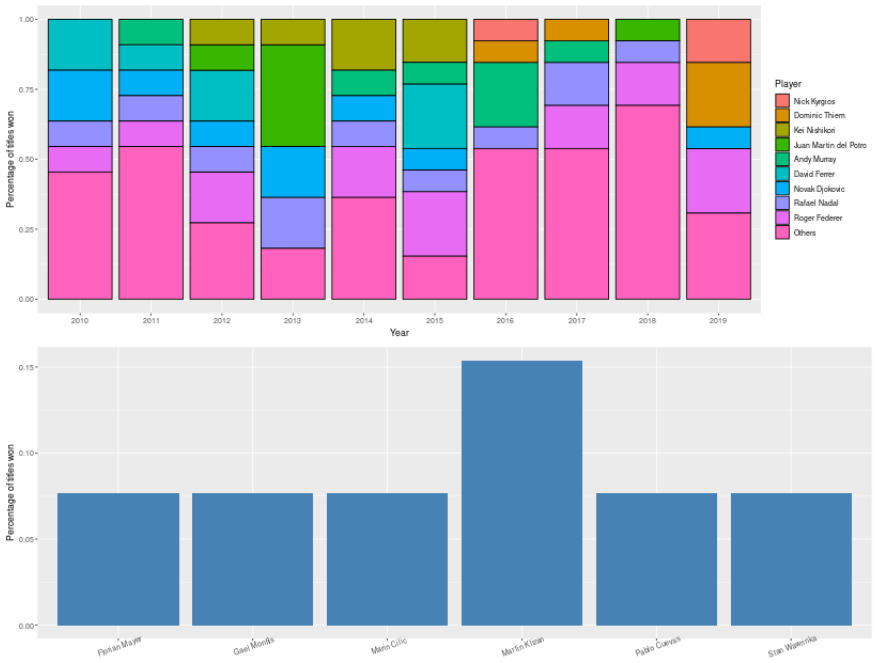

Figure 1: Stacked Bar Char of the first view for the ATP 500 tournaments in 2018

5.2 Question 2

In this visualisation, we want to represent an indeterminate number of one-to-many relationships, where the latter part of the relation are years and therefore ordered. Due to this order, we have considered a line graph in which the keys are the years and the values are the values of the statistic to be the best option, allowing us to comfortably superimpose the results of each player. Another advantage of this type of view is that it allows us to see slight changes in the annual trend of the players by looking at the slope of the lines. In this context, as we want to study the evolution of the players in one statistic, the most important variables are the year and the statistic value. Therefore, the main visual channel will be attributed to these variables, which will be spatial position and inclination, using points and lines as markers, respectively. Moreover, we want to compare different lines, so it is necessary to represent each one with different colours, leading us to a second visual channel.

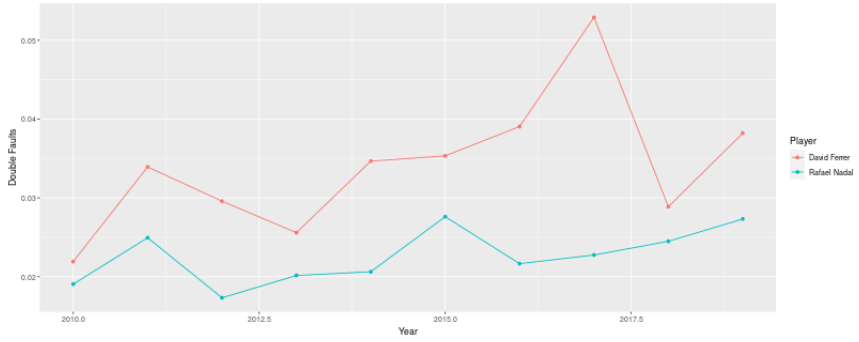

Figure 2: Evolution of double faults of Rafael Nada and Carlos Ferrer in hard and clay surfaces when ending the match as winner

As can be seen in fig. 3, this view requires 4 different parameters:

1. Players: Is a list of the players that the user is interested in comparing. These players are selected via a drop-down menu which contains a list of all the players who have won a tournament. This

selection might be done using all the players which have at least a match, but we have considered that they are to many, so we have chosen only for ATP tournament.

2. Statistic: This parameter is the statistic that the user wants to compare along the years and among the players. This parameter is selected again using a drop-down.

3. Surface: By using a group checkbox, the user can select the subset of surface to explore. The idea behind using a checkbox group widget instead a radio button widget is that we want the user to be able to select two different surfaces at the same time, and radio buttons do not allow this.

4. Total-Winner-Loser: This parameter is selected using a radio button widget and is used to specify whether the user wants to make the comparison on the total number of matches, when the player ends up winning or when the player loses. Note that the checkbox group can represent this parameter too, but we wanted to show an option for selecting all the matches.

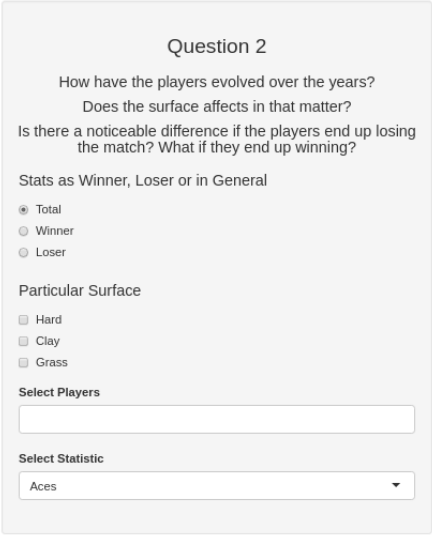

Figure 3: Question 2 parameter selectors behaviour

To finish this visualization, we want to talk about how the selected parameters are used to make the necessary transformations to display the desired information. First, we need to reduce the by filters the dataset of tournaments discarding the tournaments that take place on surfaces not selected. Then we need to filter the set of matches by keeping only the matches played (won and/or lost according to the value of Total-Winner-Lost) on the corresponding surface by at least one of the selected players. After that, all columns but the year, the statistic and the name are discarded. Secondly, we have to reduce again the previous dataframe by an aggregate that obtains the average statistic grouped by year and by player. To conclude, we need to superimpose the graphs with the previous data when displaying them, thus generating the desired graph.

5.3 Question 3

In order to answer this question we need first to summarise a lot of information for each statistic that is distributed along thousands of rows. As mentioned, to solve that the information must be concentrated with respect to two clear groups, round and type of tournament, and as we well know, one of the best approaches to concentrate and present compactly high amounts of information are the heatmaps. In fact, this approach is the best possible for this question since we have a symmetrically distributed structure for which the variation within the plot in order to find the patterns is focused on the “how-much” rather than the “what” and “where” channels. Thus, varying the magnitude of hue and luminance in concordance with the quantity of the mean statistic, we allow the users to completly focus their attention on finding clusters and outliers for which this idiom is pretty effective.

Another possible approach could have been a cluster heatmap, however, since we are working with ordinal attributes for encoding the axis, this could have led to miss perceptions about how the clusters or the outliers are distributed after the reordering.

The scalability of a heatmap is quite large, taking into account that each cell is an item, where each value can be accessed from two different keys, and that for each axis we can include an order of hundreds of attributes. However, as it was mentioned in previous paragraphs, the human visual system is a bit more limited in that aspect. That said, this solution counts with 5 attributes for the x axis, where we thought that it could be clearer for the user to associate the implicit order of the rounds, and from 3 to 4 for the y axis, where the tournaments are situated. Therefore, in some way encoding the lower levels, i.e., "Round of 32" and "ATP 250", to the bottom left corner, which is typically the origin of coordinates for any line chart or scatter plot.

With respect to the number of attributes, it must be said that we are not including all the rounds and all the types of tournaments, because they are not the same in all cases. For example, for the rounds we decided to use from the "Round of 32" onwards since the ATP 250 tournaments do not have "Round of 64". Similarly, we excluded any round apart from those, because they were either not from the tournament indeed, or not relevant for this analysis which is not considering specific 3rd and 4th position, olympic or classification matches. On the other hand, the Masters 1000 tournaments are never played on grass surfaces, therefore, they were excluded when filtering for that surface alone.

To end with the visual encoding part, we have selected a "YlGn" colormap for encoding the quantity of the statistic, see fig. 4. The main reason behind this selection, was a matter of colour association with the sport since tennis balls are green and the courts are typically grass (green) or clay (orange). Of course, courts can also be hard (blue), but by selecting this colormap, we can better relate not only the sport, but also the quantity of the statistics to a clearly differentiated variation of saturation, which also has a meaning considering that the quantity is positive and has an increasing order. The range of the colormap is adaptive for each statistic so that the main differences from each round and tourney could be clearly seen, where the yellow colour represents a lower percentage than the dark green.

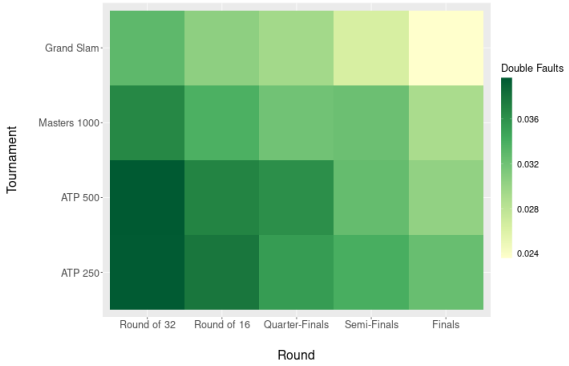

Figure 4: Heat map with the dstribution of the double faults over the rounds of tournaments and tournament types

Moving to the interaction, the user will be able to select any of the most relevant statistics of a match such as the number of aces or the service points won, being also able to also filter by one or several surfaces for a more specific analysis, see fig. 5. Note that when no surface is selected, no filtering is applied in that aspect and, therefore, the resulting heatmap will consider all surfaces.

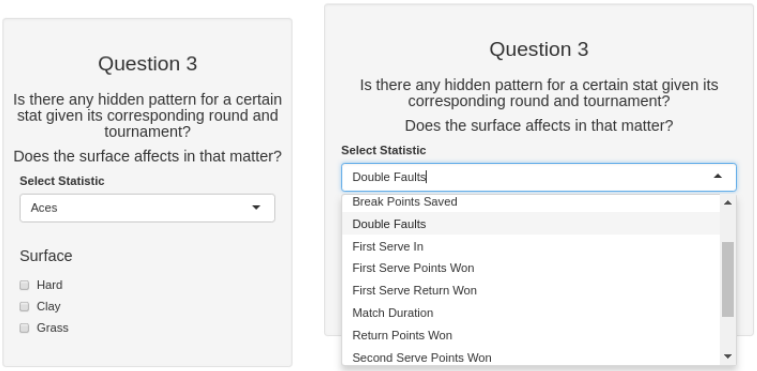

Figure 5: Behaviour of the selector of the third visualization

6 Algorithmic implementation

6.1 Question 1

The first step in the final implementation is to drop all columns that are not necessary for this task of our dataframe, in fact, we just select *tourney_type*, *year* and *singles_winner_name* columns and a list of all the possible years is created to be used afterwards.

The next parts of the algorithm are executed every time the user makes an interaction, since it needs the selected values of the different elements of the ui. Once we know the category of tournament, the algorithm filters the dataset by that category, and afterwards it performs a count operation of the number of tournaments won, grouped by the players to have an ordered list of the players that have won any tournament and the number of tournaments won by him. This is necessary to create a list of the players that are going to be shown (the ones that have won the most) and those that fall in the category of "Others".

Once that is done, the dataset that was already filtered by tourney type is now filtered by year, and we perform a count operation to determine the number of tournaments won by each player that year. With that information we construct iteratively every year the final dataframe with the winners of that year and the amount of tournaments corresponding to the category of "Others". Afterwards the data for each year is normalised diving by the total of tournaments of that year in order to get the percentages.

Finally, the normalised stacked bar chart is plotted with the information obtained for each year.

6.2 Question 2

This second visualization is implemented via 3 steps.

1. Filtering: Once the statistic, the list of players and the context (the surface and the Total-Winner-Loser) are defined, it is necessary to filter the dataframe. For each player P, those filter are the following:

- First we need to remove all the unneeded columns of the dataframe of matches, leaving us only with the *year*, *winner_pctg_STAT*, *loser_pctg_STAT*, and *tourney_name* of those matches winned or losed by *P*.
- The second filter is to get all tournaments which surface is one of the selected by the use.
- Finally, we need to remove matches from the product of the first filter whose tournament is not one of the obtained by the second filter.

2. Aggregating: Once we got the information we want for one player, we can aggregate the information by year obtaining the plot for player *P*.
3. Iterating: repeat the previous step to generate different dataframes, join the columns by year

Once this dataframe is produced, it is ready to be plotted.

6.3 Question 3

The algorithmic implementation of the last question requires four main steps. As it was mentioned in previous paragraphs, first, we must join the two datasets and for that, we are using the merge function with the tourney identifier. Once we have an unique dataframe, we can select from that the attributes on which we are more interested in, i.e., the type of tourney, the round name and the tourney surface. With respect to the statistic, that one is selected depending on the user's interaction. This step is pretty general and costful for the visualisation, therefore, is obtained beforehand out of any function, together with the definition of the lists of statistics, rounds and tournaments. After that, the remaining steps are part of a set of functions that are in charge of the filters and necessary calculations for each heatmap.

The next step takes the merged dataframe and concatenates within the row axis both the winner and loser chosen statistics with a mapper function, filtering also all the rounds and ATP tournaments out of the predefined lists. Then, in a different function this dataframe is also filtered for the selected surfaces.

Finally, the last function calls to these two previous ones, and makes the aggregation of the stat for round and tourney, resulting in a dataframe whose groups items (i.e., round and tourney columns) have to be encoded from 1 to its corresponding list lengths (5 for the round, 4 for the tourney), that, to allow the ggplot function to correctly set the implicit order of the heatmap. At this point, we have all we need for the visualisation, and what lasts is just the User Interface of the app and the configuration of the colours, labels, and sizes of the plot.

7 Conclusions

To conclude, we can summarise the different sections of the whole project by simply answering the What, Why and How questions of the different proposed solutions

7.1 Question 1

What?	Table dataset: Number of tournaments won by every player (quantitative attribute), Players (categorical attribute) and year (ordinal).
Why?	Action: Discover new information, using a browse search and a summarising query. Target: The part-to-whole relationship of the title winners and its evolution.
How?	Using a normalised stacked barchart, with spatial position channel for years, length channel for percentage of titles won and colour for players.

We can validate the use of this visualisation with an example in which it helps us to answer our question.

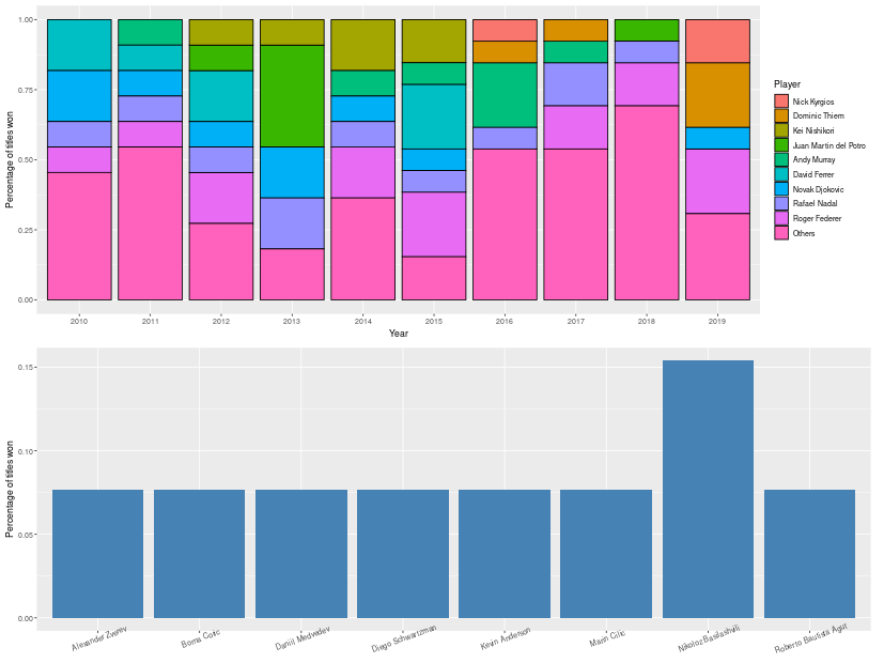

Figure 6: Distribution of all the winners of ATP 500 tournaments in 2018

In fig. 6, we see the evolution of the part-to-whole relationship for the ATP 500 titles, where we can appreciate that for the period 2012-2015 and 2016-2019 there was a higher dominance of a smaller group of players by the fact that the column of "Others" is clearly smaller in the periods 2010-2011 and 2016-2019, indicating that in these other periods the competence was higher to win these titles.

7.2 Question 2

What?	Table dataset 1: Name of all tournaments(categorical attribute), surface (categorical attribute). Table dataset 2: Name of the players, Information about the statistics achieved by each player at each match of the first dataset.
Why?	Action: Discover new information, using a browse search and a comparation query. Target: Comparation of the evolution of the different players in different statistics with different contexts
How?	Using a multiple lines plot, for a list of players selected and a statistic, superimposing the graphs and comparing them by position, using one color for each line graph

We will use the hypotheses mentioned above to show that this visualisation is useful.

A well-known hypothesis is that Rafael Nadal plays longer matches than other players like Roger Federer or Novak Djokovic. If we look at fig. 7, we can see that indeed the length of his matches is generally longer.

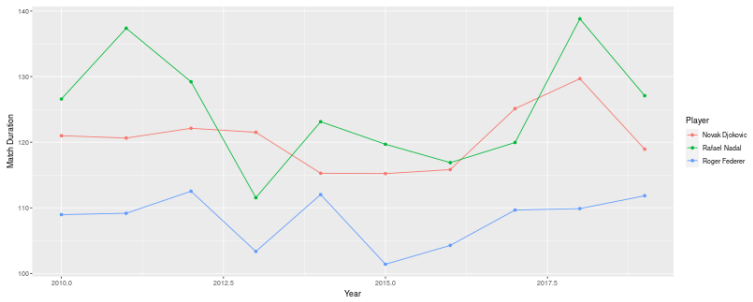

Figure 7: Evolution over the years of the match lenght of Rafael Nadal, Roger Federer and Novak Djokovic.

This leads us to a second less obvious hypothesis, as a strategy to tire your opponents is to try not to play very fast points, in particular not to score an Ace. Does Rafael Nadal risk less on his serves? If we now look at the fig. 8, we can see how Rafael Nadal's game relies less on his serving ability in general.

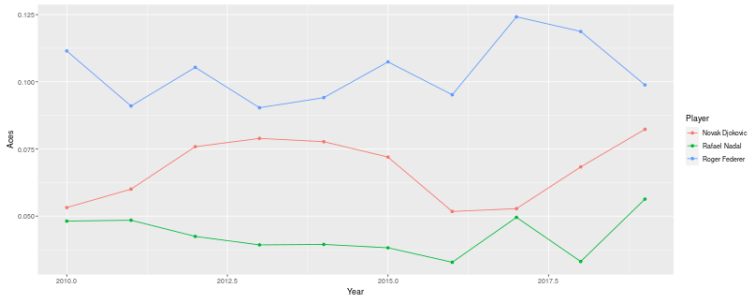

Figure 8: Evolution over the years of the percentage of Aces of Rafael Nadal, Roger Federer and Novak Djokovic.

As a final note on this visualization is that the current implementation requires to traverse the match dataframe once for each player, when it is possible to traverse it only once. A new implementation we have thought would be to save a column with the player's name and when aggregating, aggregate by

year and player, thus going through the dataset many fewer times. We also thought that this is not a very big problem, as the number of players to be compared should not be too high if the graph is to be understood, but we felt that it should be explained.

7.3 Question 3

What?	Table dataset 1: Number of tournaments won by every player (quantitative attribute), Players (categorical attribute) and year (ordinal). Table dataset 2: Information about the statistics achieved by each player at each match of the first dataset.
Why?	Action: Discover new information, using a browse search and an identify one query. Target: The possible clusters, outliers and patterns, shown from an average statistic at each round and tourney
How?	Using a heatmap, for the list of ATP Tournaments and minimum common tourney rounds, and taking advantage of a suitable colour channel to represent the quantity of the stat and the context of the data.

Here we can, for instance, answer the question for validating another hypothesis, which is that the overall of players in a Grand Slam is much better than those from an ATP 250, see figure fig. 9

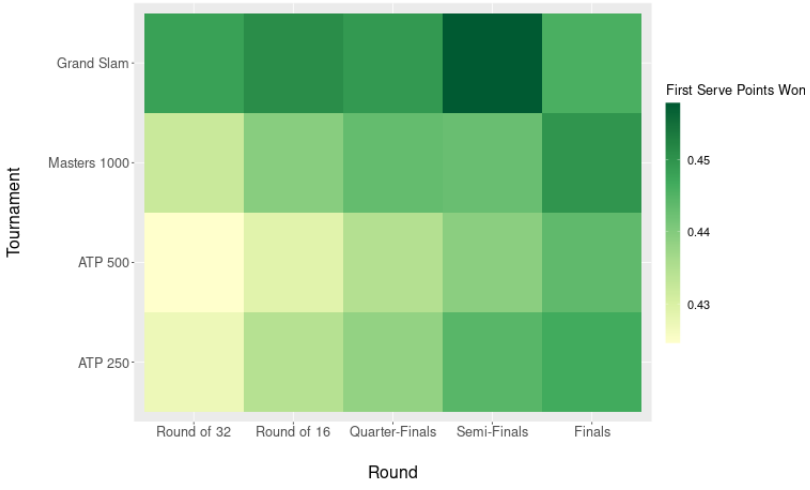

Figure 9: Heat map with the distribution of points won using only the first serve.

In this visualisation, that statement is very clear. We can see that the average number of first serve points won in finals or Grand Slams is higher than in Rounds of 32 of an ATP 250, which implicitly is confirming that the overall level in those situations is usually better. This way, we could generate another hypothesis that could focus the attention in the different tournaments of a specific type to see whether there are specific ATP 250 or ATP 500 tournaments where the difference is also noticeable.