Università di Pisa

Computer Science Master Degree

Tennis Matches Data Analysis

Authors: Dario SALVATI,
William SIMONI and Andrea ZUPPOLINI

ACADEMIC YEAR 2020/2021

Contents

1 Introduction

The purpose of this project is performing a data mining study on the *tennis matches* dataset in order to extract relevant features from the data. Using various approaches, we want to build a useful set of statistics which weren't originally available in the dataset but that help understanding and working with the data provided.
This work is mainly going to focus on tennis player's profile providing useful features to be used for further purposes and it's divided into three different phases:

- **Data understanding**, in this phase we use different plots, algorithms and statistics to determine the quality of the data, detect outliers and missing values, correlation between attributes and more;
- **Data preparation**, here we use the information retrieved from Data understanding phase to operate on the data. The typical operations are correction of outliers and filling of the missing values (where possible);
- **Data integration**, we integrate the original data with new features then we perform data understanding on the just created attributes;
- **Clustering**, using clustering algorithm we can study the data and extract new information;

Please, consider that the plots that are going to be shown below represent just a relevant part of the whole attributes. Since their number is considerable, we preferred to show less plots in order to make this report more readable and all the diagrams omitted are available in the material provided in addition to this report.

2 Data understanding

This section is dedicated to the data understanding part of the project. Here the aim is getting to know the data, analyze its structure and manage possible missing values or outliers.

2.1 Dataset Description

Before working on the three datasets, let's first understand the semantic of each attribute and their possible values. The `male_players.csv` and `female_players.csv` are composed of patterns that are characterized only by the name and the surname of the athlete. The `tennis_matches.csv` dataset instead is composed of patterns characterized by the following attributes:

- **tourney_id** [object], a unique identifier, different for each tourney, that also specifies the year in which the tourney has taken place;
- **tourney_name**[object], the name of the tourney;
- **surface** [object], the type of surface on which the match was played. This attribute is categorical and can assume values *Hard, Clay, Grass, Carpet*;
- **draw_size** [float64], indicates the number of players in the draw, sometimes rounded to the nearest power of 2;
- **tourney_level** [object], indicates the type of tourney. There's a distinction between male tournaments and female tournaments: male tournaments can be *Grand Slams*, *Masters 1000s*, *other tour-level events*, *Challengers*, *Satellites/ITFs*, *Tour finals and other season-ending events* and *Davis Cup*; while female tournaments can be *Premier*, *Premier Mandatory* and *International*;
- **tourney_date** [float64], date of the tourney, in the format YYYY/MM/DD;
- **match_num** [float64], specific identifier of a match in a tourney. It can start counting up from 1 or counting down from 300;
- **winner_id** [float64], an unique identifier used to identify the winner;
- **winner_entry** [object], this attribute specifies the entry of that player in the tourney, it can assume *WC* meaning wildcard, *Q* which stands for qualifier, *LL* indicating lucky losers, *PR* for the protected players, *ITF* for the ITF players. There're few more random values that rarely occur;
- **winner_name** [object], name and surname of the winner of the match;
- **winner_hand** [object], the hand used by the winner of the match. It can assume values *Right*, *Left*, Unknown;
- **winner_ht** [float64], the height of the winner of the match;
- **winner_ioc** [object], it's a three character long code used to indicate the nationality of the winner of the match;
- **winner_age** [float64], the age of the winner of the match;
- **loser_***, all the attributes specified above for the winner of the match are also replicated for the loser of the match;
- **score** [object], final score of the match;

- **best_of** [float64], indicates the maximum number of sets in order to win the match;
- **round** [object], represents the stage of the tournament, for example SF stands for semifinal and F for final;
- **minutes** [float64], the length of the match in minutes;
- **w_ace** [float64], the number of aces performed by the winner of the match;
- **w_df** [float64], the quantity of double faults of the winner;
- **w_svpt** [float64], the number of saved points achieved by the winner of the match;
- **w_1stIn** [float64], the number of first services done by the match winner;
- **w_1stWon** [float64], the number of first-serve points won by the winner of the match;
- **w_2ndWon** [float64], the number of second-serve points won by the winner of the match;
- **w_SvGms** [float64], the number of served games performed by the winner of the match;
- **w_bpSave** [float64], the number of breakpoints saved by the winner of the match;
- **w_bpFaced** [float64], the number of breakpoints faced by the winner of the match;
- **l**, all the attributes specified above for the winner of the match are also replicated for the loser of the match;
- **winner_rank** [float64], the rank of the winner of the match;
- **winner_rank_points** [float64], the number of ranking points of the winner of the match;
- **loser_rank** [float64], the rank of the loser of the match;
- **loser_rank_points** [float64], the number of ranking points of the loser of the match;
- **tourney_spectators** [float64], the amount of spectators watching the tourney;
- **tourney_revenue** [float64], the overall revenue of the tourney;

2.2 Data Quality

The next step is to assess the quality of the data provided.
The first thing that we did was computing the percentage of missing values for each attribute. The attributes *winner_rank, winner_rank_points, loser_rank, loser_rank_points* have a percentage of missing values between 10% and 20%, while the attributes *w_ace, w_df, w_svpt, w_1stIn, w_1stWon, w_2ndWon, w_SvGms, w_bpSave, w_bpFaced* and their l_* respective have a missing value percentage around 55%. The most considerable number of missing values is in the attributes *winner_entry, winner_ht, loser_entry, loser_ht*, that have more than 73% of missing values. The remaining attributes have a percentage of missing values which is too small to be considered a relevant issue. Also, no exact duplicates have been encountered during this analysis.

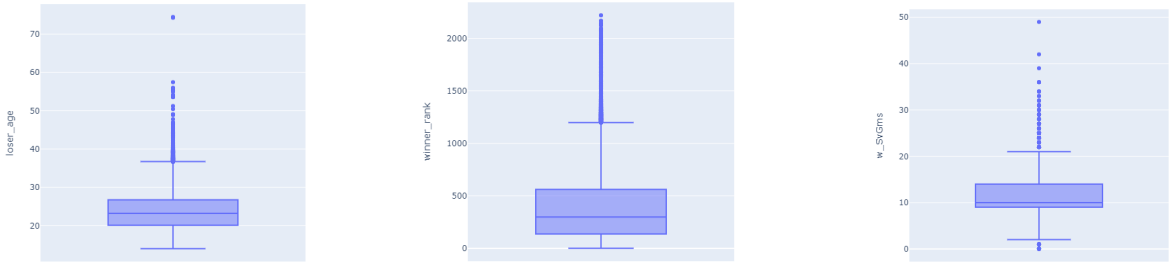
2.3 Data Analysis

Data analysis is a subtask of data understanding. In this phase we visualized the data distribution, in order to find errors, outliers and so on. Please notice that here we will show only a pool of the whole set of attributes. For the analysis of the remaining attributes, please check the notebooks we provided.

2.3.1 Outliers/Error detection

We performed data understanding using *boxplots* and *histograms*.

The boxplots are particularly useful to detect outliers. Looking at the Figure 1, we noticed that several points are out the two horizontal lines indicating the noise limit of the distribution. Those two lines are calculated starting from the inter quartile range (IQR), giving us a tool to detect outliers.

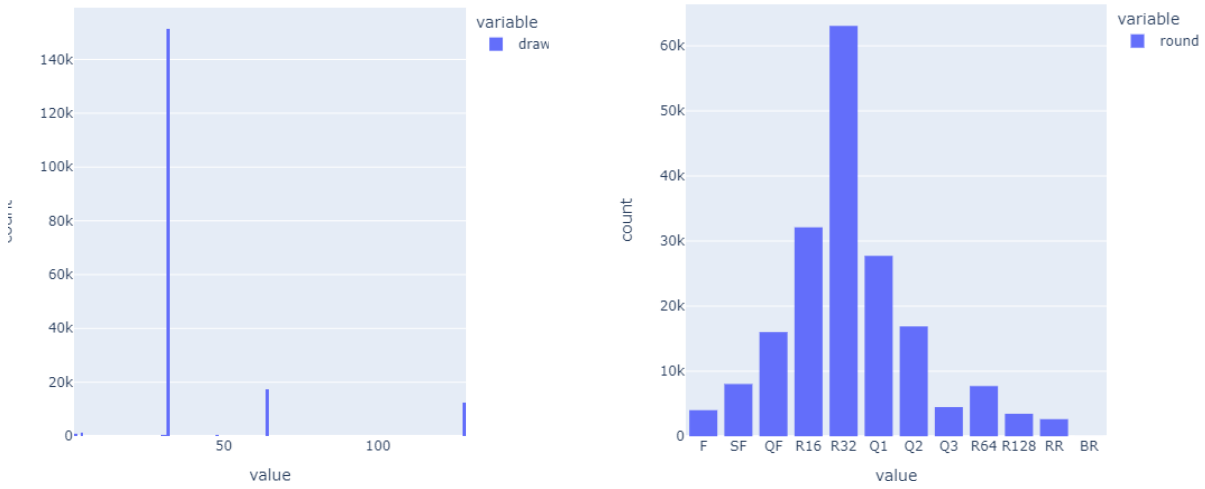


(a) (b) (c)

Figure 1: Box plots for the attributes: (a) loser_age; (b) winner_rank; (c) w_svgms.

Histograms allow us to compare our expected distribution of an attribute with respect to the real one. Moreover, as it will be described later, we were able to detect errors by

only looking at it.



(a) (b)

Figure 2: Histograms for the attributes: (a) draw size; (b) round;

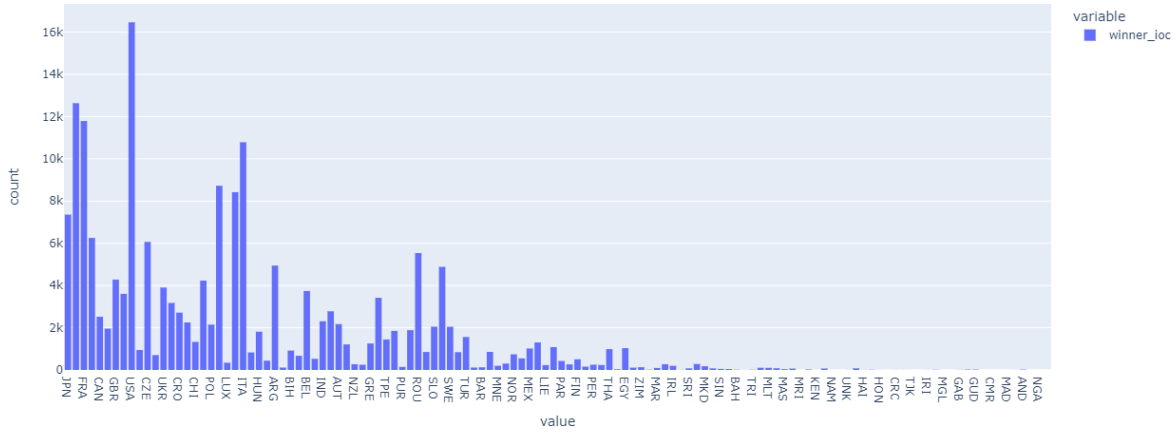


Figure 3: Histogram for IOC

2.3.2 Correlation analysis

In the following subsection we presented the correlation analysis on all the attributes, underlining high values of correlation.

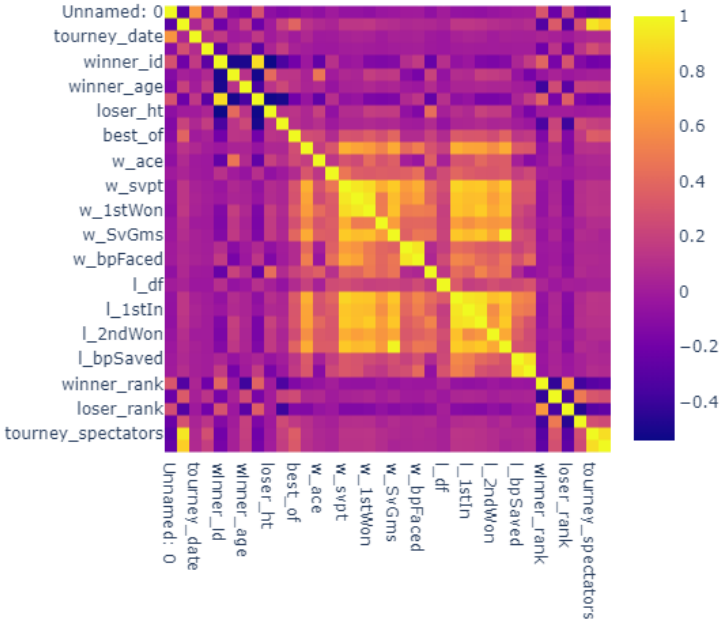


Figure 4: Caption

The correlation matrix shows the following correlation results:

- Between **draw_size** and **tourney_spectators** there is a correlation of 0.93, that makes sense because the higher is the number of participants, the higher will be the number of match and therefore of spectators;
- **Tourney_revenue** and **tourney_spectators** have a reasonable correlation of 0.9, probably because the revenue is linked to the amount of spectators;
- The correlation between **draw_size** and **tourney_revenue** is 0.84, the bigger is the **draw_size** of the tourney the bigger is its revenue;
- For both winner and loser there's correlation between the number of breakpoints faced **W_bdFaced**, **L_bpfaced**, and the number of breakpoints saved **W_bdSaved**, **L_bdSaved**;

3 Data preparation

3.1 Missing values

We managed missing values in different ways:

- If we had sufficient information, we tried to compute the actual value. For instance, we filled in the missing values of most of the players' ages by searching another match with both the player's age and the tourney date not null. From these two pieces of information, we computed the players' birthday, and thus we finally were able to determine their age for a given match given the tourney date of that match. By this process, we deleted approximately 1500 missing values.
- If we did not have enough information, we substituted merely the nan value with 'U' for the categorical data and '-1' for the numerical data.
- Finally, for other attributes such as w_ace or w_df, we did not perform anything, taking care of excluding them from further analysis.

3.2 Outliers

As introduced above, the outliers detection was based on the boxplots and histograms. We used information from those plots to delete outliers. We set to -1 or 'U' all the attribute values above the upper fence or below the lower fence given by the box plot. For instance, we assigned -1 to all the heights above 206 cm and below 157 cm, which are the upper and lower fences of the height attribute.

3.3 Errors

Looking at the histograms, we found out several errors. Regarding the draw size attribute, we approximated it to the nearest bigger power of 2, as it usually is in the draws. For instance, a draw size of 29 becomes 32. For the round attribute, we considered "BR" as a typo for "RR", and therefore we substituted each occurrence of "BR" with "RR".

3.4 Attribute Deletion

We decided to delete winner_rank_points and loser_rank_points for two main reasons:

- there are already two attributes, winner_rank and loser_rank, give us similar information.
- data distribution does not hold significant information

4 Data integration

In this section we describe the data frames used in the later sections for clustering and classification. We created different dataframes to have multiple points of view on the dataset:

- **matches_male_players**, and its female counterpart **matches_female_players**, provide a general profile describing some useful features of the players;
- **player_match_statistics**, a dataframe describing more specifically statistics on a small set of players;
- **tourney_statistics** gives a more precise view on the tourneys.

4.1 Matches dataframe

The matches dataframe contains 3019 rows for the male section and 7062 for the female section. This is an example entry of those dataframe:

	Name	Sex	Hand	IOC	Won_Tournaments	Number_of_Matches	Win_Percentage	best_year_matches	worst_year_matches	best_year_wins	worst_year_wins
0	Mirjana Lucic	F	R	CRO	0.0	88.0	0.534091	2016.0	2018.0	2016.0	2018.0

Figure 5: Example entry from players dataframe

As shown above, the new attributes are:

- Won_tournaments, number of won tournaments;
- Number_of_matches, number of played matches;
- Win_percentage, percentage of won games;
- BestWorst_year_matches, the year with the highestlowest number of played matches;
- BestWorst_year_wins, the year with the highestlowest number of won matches;

4.2 Player_game_statistics

This dataframe contains 2232 rows of players statistics, to have closer view on the various game styles. The new attributes are:

- ace_percent, the percentage of ace done by the player;
- break_point, the percentage of break points done by the player;
- fspw, the number of first serve points won by the player;
- average_rank, showing the average rank the player had over all the tourneys;

	Name	ace_percent	break_point	fspw	average_rank
1	Gonzalo Villanueva	0.034551	0.546371	0.645863	500.761905
2	Ben Patael	0.061041	0.581081	0.656706	466.525000
3	Filipe Brandao	0.032609	0.500000	0.519231	380.000000

Figure 6: Example entry from Player_game_statistics

4.3 Tourney_statistics

This other dataframe shifts the focus on the tourneys. There we find new features describing various aspects of the competitions. The following attributes were introduced:

index	tourney_name	tourney_id	avg_spect	avg_rank	avg_revenue	tot_match
0	Brisbane	2019-M020	2366.0	129.734694	449800.02	49
96	Doha	2019-0451	2773.0	180.406250	563339.81	48
139	Pune	2019-0891	447.0	104.125000	87559.10	16

Figure 7: Example entry from Tourney_statistics

- avg_spect, the average number of spectators of the tourney;
- avg_rank, the average rank of the tourney participants;
- avg_revenue, the average revenue of the tourney;
- tot_match, the overall number of matches played during the tourney;

5 Clustering

Having many data frames to cluster, we first defined for each clustering method a general methodology to find the optimal parameters. The general idea is first to find a bunch of valid parameters by a grid search based on the Silhouette score and then see visually which of these parameters lead to the best clustering in terms of interpretability. For each clustering method, we used the implementation provided by the Sklearn library.

5.1 Preprocessing

We have done most of the work in the data understanding and data integration phases. In this phase, we just scaled the data using two different types of algorithms depending on the data frame:

- StandardScaler: standardize features by removing the mean and scaling to unit variance.

- RobustScaler: scale features using statistics that are robust to outliers.

Table 1: Scaling algorithm used for each data frame and clustering method. R stands for RobustScaler while S stands for StandardScaler

dataset	Kmeans	DBSCAN	Hierarchical
matches_male_players	R	S	S
matches_female_players	R	S	S
player_match_statistics	S	S	S
tourney_statistics	S	S	S

5.2 KMeans parameters

The most significant parameter in Kmeans is the k that determines the number of clusters the algorithm will produce. To find a good k, we performed a grid search over possible values of k, and finally, we plotted the silhouette scores and the inertia for those values. In the silhouette plot, we searched for the k that maximizes the score. While in the inertia plot, we sought for the k in the elbow of the curve. We used this approach to find suitable candidates for the k parameter. Eventually, we plotted the Kmean result using those ks, and we chose the k that gives the best interpretability.

For instance, in the matches_male_players dataset, the resulting curves were 8a for the silhouette score and 8b for the inertia. Based on these curves, we chose to try with a k close to 5.

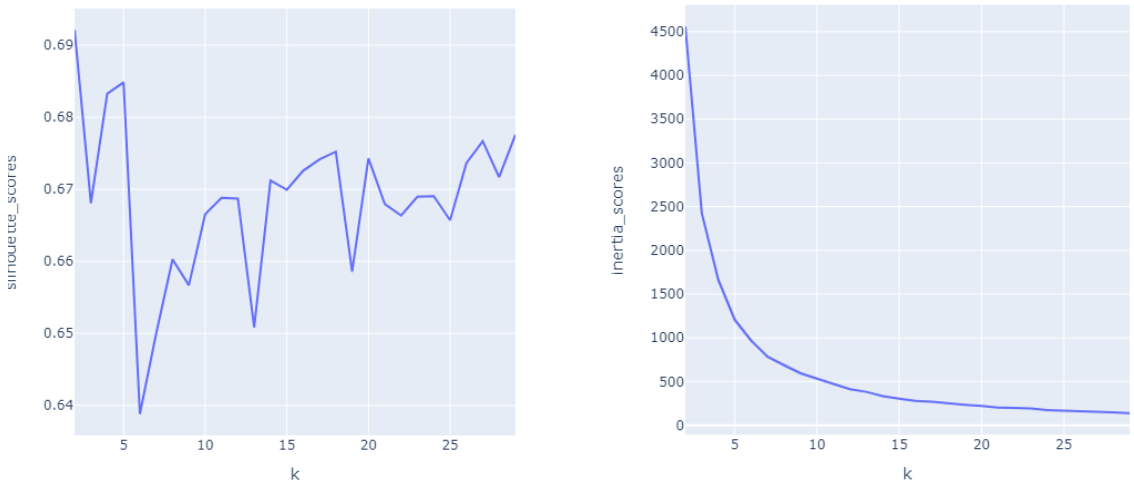
Table 2 shows, for each data frame, the k we finally chose and the corresponding silhouette score and inertia.

Table 2: Kmeans table

dataset	k	Silhouette score	inertia
matches_male_players	5	0.685	1,211
matches_female_players	4	0.619	3,778
player_match_statistics	3	0.319	5,018
tourney_statistics	4	0.338	6,314

5.3 DBSCAN parameters

The most significant parameters in DBSCAN are:



(a) (b)

Figure 8: (a) Silhouette scores and (b) Inertia values for Kmeans in matches_male_players dataset

- epsilon: the maximum distance between two samples for one to be considered as in the neighborhood of the other.
- min_samples: the number of samples (or total weight) in a neighborhood for a point to be considered as a core point. This includes the point itself.

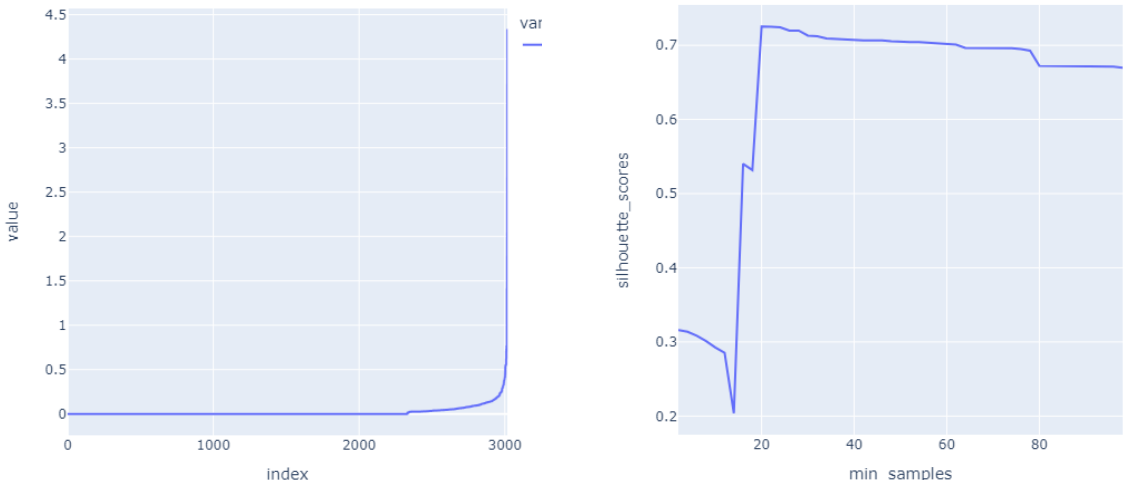
To select a good epsilon, we compute the distance from each point to its closest neighbor using the NearestNeighbors. Then we sorted the resulting distances and plotted the result. Finally, the optimal value for epsilon is at the point of maximum curvature. For instance, in the matches_male_players dataset, given the plot shown in Figure 9a, we choose for $\epsilon = 0.35$.

Fixed the ϵ, we performed a grid search over possible values of min_sample computing for each try the silhouette score. The resulting plot for the matches_male_players dataset is shown in Figure 9b. Eventually, we chose the min_sample that maximizes the score, and thus in the matches_male_players dataset, that value is 28.

Table 3 shows, for each data frame, the parameters we finally chose and the corresponding silhouette score.

5.4 Hierarchical clustering parameters

For the hierarchical clustering we followed again a grid search approach based on the Silhouette score. For the hierarchical clustering we identified two key parameters:



(a) (b)

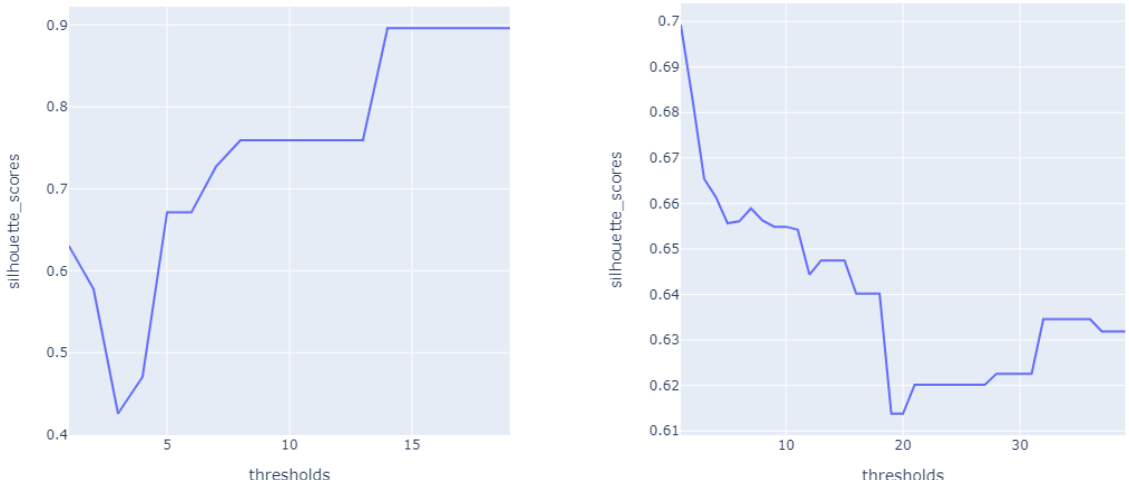
Figure 9: (a) distances and (b) Silhouette scores for DBSCAN in matches_male_players

Table 3: DBSCAN table

dataset	ϵ	min_sample	Silhouette score	number of clusters
matches_male_players	0.35	22	0.664	(1 + noise)
matches_female_players	0.2	30	0.625	(1 + noise)
player_match_statistics	0.6	15	0.27	(1 + noise)
tourney_statistics	0.45	30	0.14	(2 + noise)

- linkage criteria: determines the metric used for the merge strategy. In detail, we consider the ward and the complete criteria.
- distance threshold: threshold above which, clusters will not be merged.

Then, we visually evaluated the hierarchical clustering over the data frames using the parameters suggested by the grid search. Figure 10 shows the resulting curves of the grid search with the data frame matches_male_players.



(a) (b)

Figure 10: Silhouette scores with an increasing distance threshold and with linkage criteria (a) complete or (b) ward in the data frame matches_male_players

Table 4: Hierarchical table

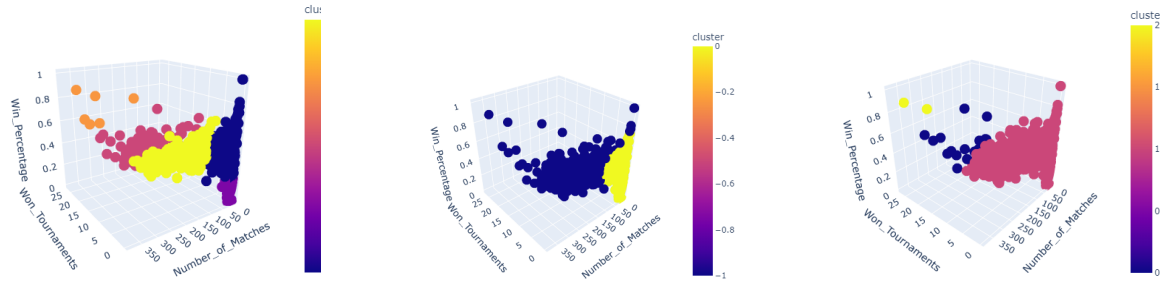
dataset	linkage	distance threshold	Silhouette score
matches_male_players	complete	10	0.65
matches_female_players	complete	10	0.67
player_match_statistics	complete	10	0.42
tourney_statistics	ward	70	0.29

5.5 Results

This section presents the clustering results.

5.6 matches dataframe

As we can notice in Figure 11, all the methods are able to discriminate between strong and weak players. For instance, in the Kmenas result, the orange cluster represents the top-tier players while the purple cluster represents players that have not disputed many games.



(a) (b) (c)

Figure 11: Clustering results for: (a) kmeans with k=5; (b) DBSCAN; (c) hierarchical clustering.

5.7 Player_Game_statistics

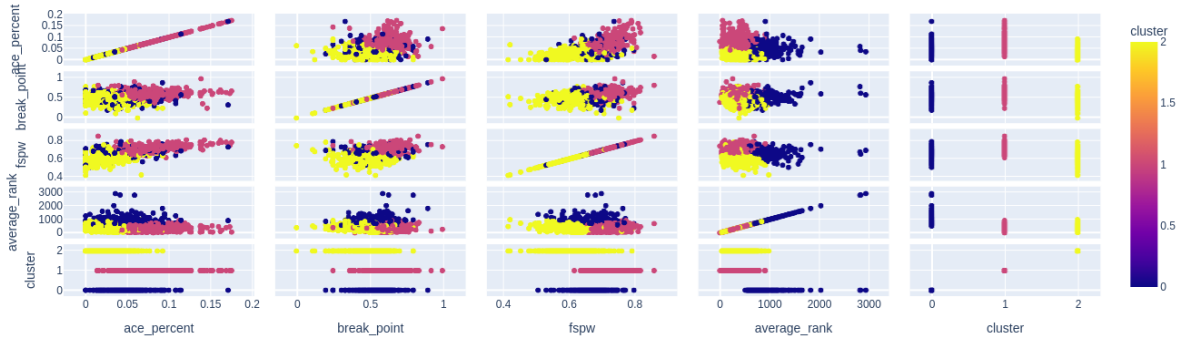


Figure 12: K-means clustering with k=4

5.8 Tourney_statistics

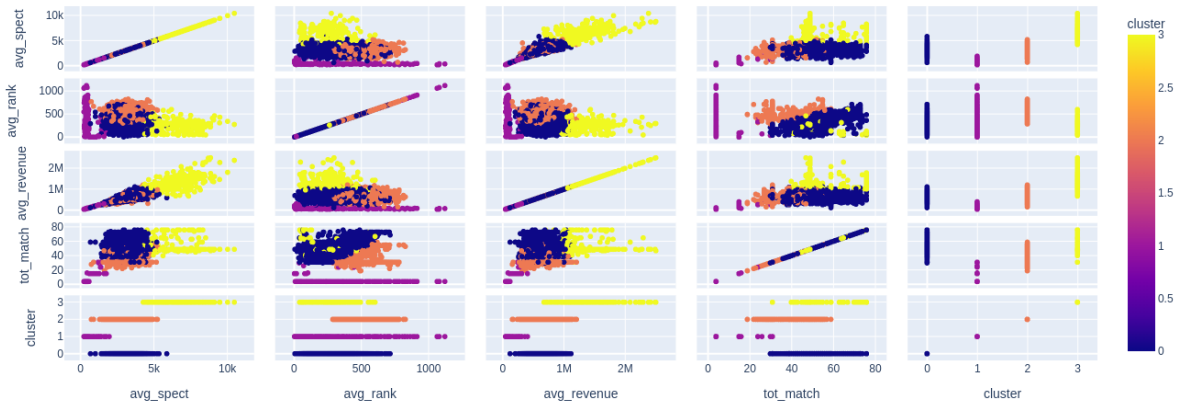


Figure 13: K-means clustering with k=4