Dipartimento di Informatica

Corso di Laurea Magistrale in Informatica

Data Mining: Tennis matches

Antonio Zegarelli, Simone Rizzo

Groupd ID: 1

Anno Accademico 2021/2022

Contents

1 Introduction **1**

2 Data Understanding **1**

2.1 Data semantics 1

2.2 Data integration 2

2.3 Cleaning and fixing 2

2.4 Correlation 4

3 Feature engineering **4**

3.1 Matches 5

3.2 Players 5

3.3 Overall players statistics 6

3.4 Correlation 7

4 Clustering Analysis **8**

4.1 Normalization 8

4.2 Feature selection 8

4.3 K-means 9

4.4 DB-scan 11

4.5 Hierarchical 13

4.6 Alternative clustering techniques 15

4.6.1 X-means 15

4.6.2 Fuzzy-c-means 16

5 Classification **17**

5.1 Results 18

6 Explanation Analysis **19**

6.1 SHAP 20

6.2 LIME 21

6.3 LORE 22

6.4 Evaluation 23

7 Conclusion **24**

1 Introduction

The purpose of this report is to carry out an exploratory analysis based on data mining tools on the tennis match dataset. The main stages of the analysis are:

- Data Understanding and Preparation: analysis and cleaning of the datasets.
- Feature engineering: derivation of new features and elimination and/or collapse of related ones.
- Clustering: identification of groups of players with similar properties to try to build a profile.
- Predictive Analysis: labeling and prediction of high and low ranked players
- Explanation Analysis: use explanation algorithms over the created models.

2 Data Understanding

The datasets used are:

- male_players of size (55208, 2)
- female_players of size (46172, 2)
- matches of size (186128, 50)

In both male and female datasets, we have that each record represents the full name of the players, instead, in the match one we have records representing tennis matches.

2.1 Data semantics

The semantics for each feature in the matches dataframe is (we only report the ones not/not well documented):

- score: is the result of the match
- round: Round of a tournament
- w/l_svpt: total number of serve points (a shot to start a point).
- w/l_stIn: number of first serve in play.
- w/l_bpSaved: number of break points for the opponent that our player break so he saved the game.

- w/l_bpFaced: number of break points that the player has faced.
- winner/loser_rank: rank of the player, is assigned using the rank points sorted in a descending order so the first is the player with most points.
- winner/loser_rank_points: points of the player, adds up players best results in tourney rounds in the previous 52 weeks.
- tourney_spectators: the number of spectators of the tourney, not the match.
- tourney_revenue: the total revenue of the tourney.

2.2 Data integration

Since we do not have the user id on all three datasets, the player name must be used for the integration. In order to use the names, we need to normalize them, transforming them into a lower case by removing points and trailing / leading spaces. In this way, it was possible to integrate information relating to the gender of the players. Among the names, 2 types of errors were identified:

- Name present in male and female
- names not present

The resolution of these errors will be covered in **subsection 2.3**.

2.3 Cleaning and fixing

Duplicates Starting with the removal of duplicates on matches, 309 records have been eliminated.

Missing Values Displaying the missing values in **Figure 1**, it can be seen that the part relating to the match statistics ranging from minutes to l_bpFaced is missing in 1/3 of the records. Since you want to create profiles of the various players their statistics are necessary information, for this reason, the empty lines have been eliminated. The other most empty columns are winner/loser_entry and winner/loser_ht.

Gender Analyzing the names for which it was not possible to find a match, it is noted that the problem is due to errors in the name. Using the Levenshtein distance between

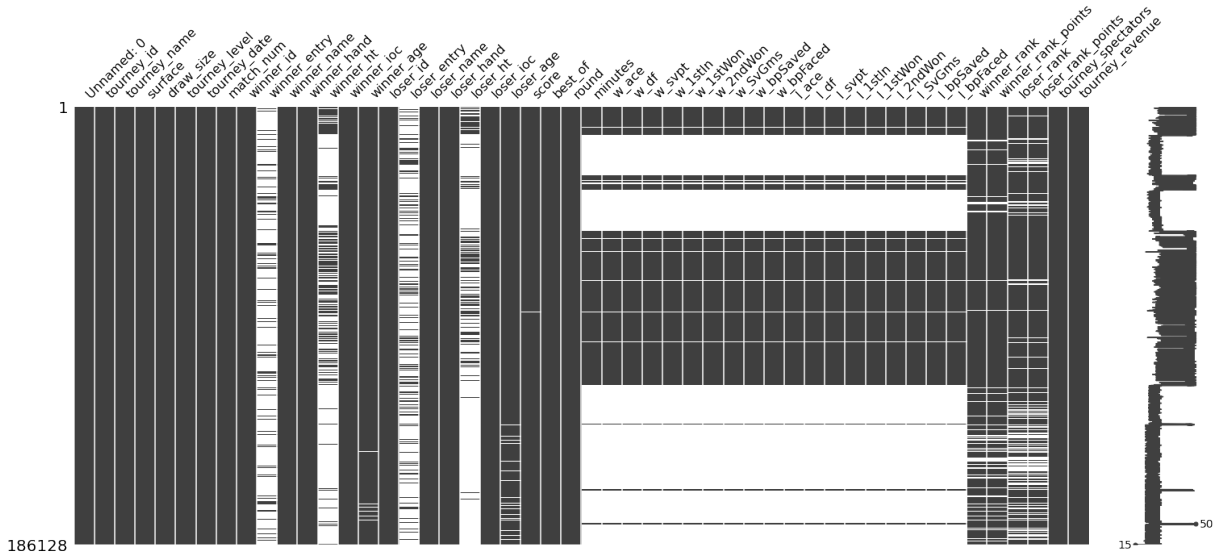


Figure 1: Missing data visualization

match names and male / female names, 26 errors were corrected.
For the names of players present in both datasets, gender discrimination was made using matches. Since from the information on the dataset we know that there are no unisex tournament matches, we can use the opponent's gender in this way 3 errors have been corrected.

Player ID Analyzing the ids of the players we note that 2 players share the same id only if they have different genders. To have unique ids, all the ids of the players have been transformed into a string and a leading zero has been added to the ids of the women.

Surface The information relating to the type of field for the match is missing in l17 records out of a total of 81982, it is therefore reasonable to carry out a sampling while maintaining the proportion.

Hand In the hand field, many players have different values in the various matches. These players have been assigned the hand class they need most. The U-value (unknown) has been entered for players who could not be awarded a hand.

Minutes Between the minutes there were outliers far outside the average and current records. In the time range of the dataset the fastest and longest game records are [20,

420] minutes. The outliers for the upper bound have been moved to it. For games with duration 0, we can exploit the score to find if it is an error, and in case replace it with the median.

Age The age-related float has been transformed into an integer. Since the time range of the dataset is 5 years, the birth year was created to represent the ages. For the cases resulting in different born dates the value that appears the most was used.

Height In the height field, 2 main types of errors are identified:

- different heights for the same player - outliers
- missing values

For the missing values, it was decided not to enter values because they represent a large portion. For the others, a manual correction was possible due to their low number.

2.4 Correlation

The correlation plot can be seen in the notebook. Most of the correlated features are those related to the statistics of players in matches.

- 1stIn - svpt - 1stwon (Positive)
- w_svgms - l_svgms (Positive)
- bp_saved - bp_faced (Positive)
- tourney_spectators - draw_size - tourney_revenue (Positive)
- minutes - svpt (Positive)

3 Feature engineering

Looking at the correlated features it is possible to reduce the dimension by eliminating/-collapsing them together. Moreover looking at ultimatetennisstatistics.com it can be seen that there are a lot of useful statistics about tennis matches, thus some of them can be created based on the features contained/inferrable in the dataset.
We can divide the features in 3 groups:

- Matches: about the matches in the dataset
- Players: about the players in the dataset
- Overall players statistics: global statistics of the players over all the matches

3.1 Matches

The score feature can be exploited to obtain more information about the match:

- Sets: winner/loser sets won
- Games: winner/loser games won
- Tiebreaks: winner/loser tiebreaks won and total tiebreaks

There are also statistics on the match that can be inferred:

- 2ndIn is the number of second serves in play calculated as svpt - 1stIn - df
- 1st serve return points won is calculated as opponent 1stIn - opponent 1stwon
- 2nd serve return points won is calculated as opponent 2ndIn - opponent 2ndwon;
- break points won is calculated as opponent bpfaced - opponent bpsaved;

3.2 Players

To better understand the information about the players that we have in our dataset we create a new dataset containing the principal information about them: Born date, height, rank min, rank min opponent, hand, gender, IOC. The unique players are 4257 divided in 30% female and 70% male. The average born date for the player is (1994-27 years old) we can see that the female are younger than males and also the heigh is different the male are higher than female and also the male rank is higher than female.

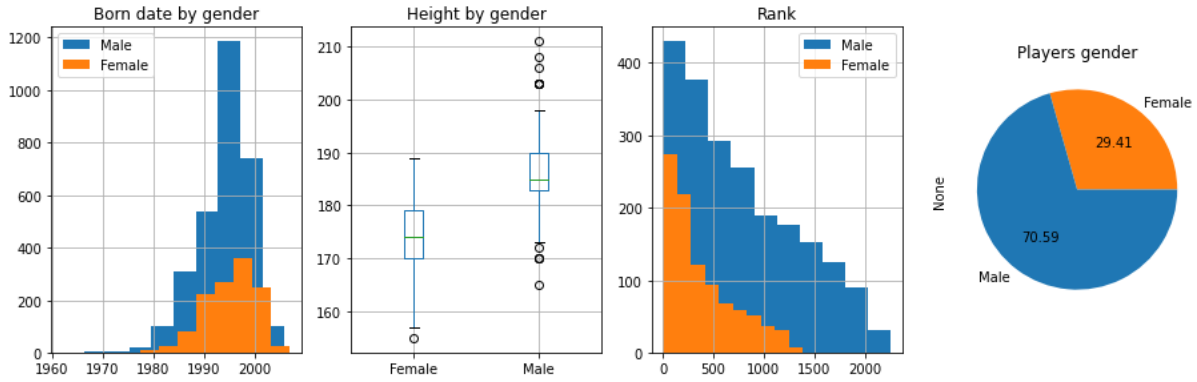


Figure 2: Unique players info

3.3 Overall players statistics

Those are the statistics about the global performance of each player over the matches in the dataset.

- total matches and total for each surface type
- win percentage on each surface type
- tourney won and total played
- first serve win percentage: 1stWon/1stIn
- second server win percentage: 2stWon / 2stIn
- ace percentage: ace / svpt
- double fault percentage: df / svpt
- average number of games per match: (games won + o games won) / n matches
- tiebreak won percentage: player tb won / tiebreaks total
- breakpoint saved percentage: player bpSaved / player bpFaced
- breaking opponent's serves percentage: opponent bpSaved / opponent bpFaced
- succesful 1st serves percentage: 1stIn / svpt
- 1st serve return points won percentage is calculated as 1st serve return points won / opponent 1stIn;
- 2nd serve return points won percentage is calculated as 2nd serve return points won/ opponent 2ndIn;
- break points converted (won) percentage is calculated as break Points won / oppo-

nent bpfaced.

- minutes per match: minutes / match_count
- tourney importance: Sum over all the matches of the tourney_revenue. It encodes implicitly the round feature (and so also the draw size), since more matches in a tourney imply that the revenue is summed more times. So it is an evaluation of the player importance based on the played tourneys.

3.4 Correlation

In **Figure 3** are shown the correlations between the features of the new dataset on the players.

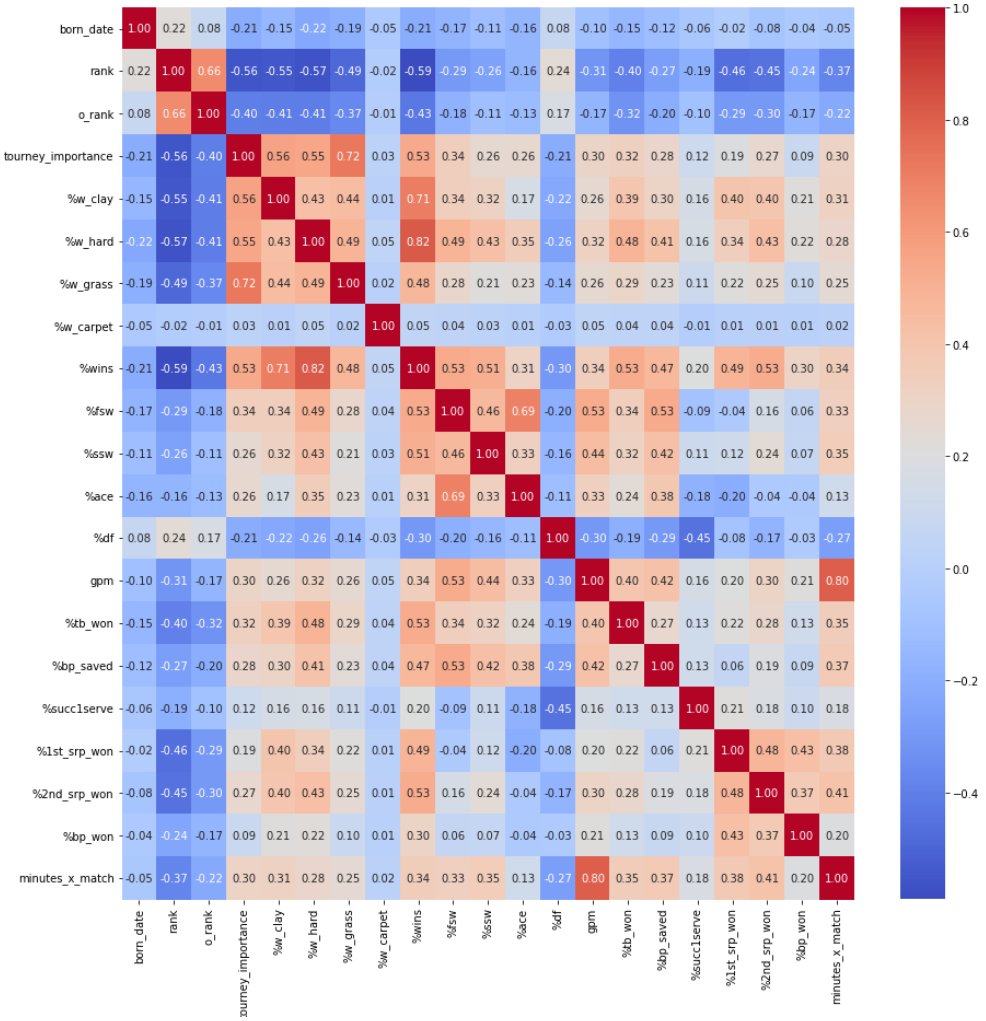


Figure 3: New features correlation

4 Clustering Analysis

In this chapter, we discuss different clustering algorithms, fine-tuning their hyperparameters and evaluating results from both the statistical and semantical point of view.

4.1 Normalization

We decided to normalize the values using the MinMax scaler since we use a lot of percentage values mixed with non-percentage, the minmax scaler will transform them between 0-1 so better visualization during cluster line plots and the other fact is that not all attributes that we used follow a Gaussian distribution (so Standard scaler isn't the best).

4.2 Feature selection

For the feature selecting phase PCA has been used. This way it was possible to detect for each component the most representing feature. By analyzing the PCA components it can be seen that the first one captures 65.7% of the variance of all the features while the second one 17.2%. Thanks to the principal component composition we found that the most important feature is the rank, which divides the points in three main groups as we can see in the 3D visualization **Figure 4**.

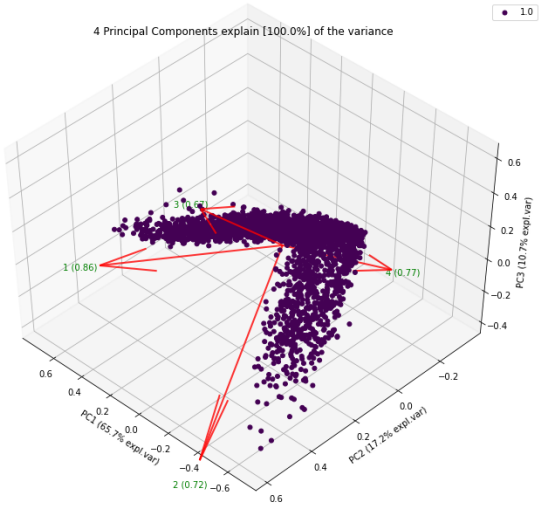


Figure 4: PCA 3D

Using those results over the PCA we designed our best features to divide the point cloud as best as possible. We have selected 4 features to perform the clustering, their correlation plot can be seen in **Figure 5**. In particular, we have ignored the categorical attributes as there isn't a proper metric to define a distance function over them. The attributes used are rank, tourney_importance, %ace, %1st_srp_won these represent, in order, the strength of the

player, the number of matches played with their importance, offensive ability, and defensive ability. The height has been excluded because only 500 has this information thus the final dataset for clustering contains 3050 unique players. For these attributes, we used the Euclidean distance as a metric function since it is more natural for this kind of attributes.

4.3 K-means

Best value of k The plot in **Figure 6** has been used to select the best k. It contains for each number of k between 2 and 20 the SSE value and silhouette score. For each try is selected the best among 10 runs (with the lowest SSE) and with a maximum of 300 iterations of the K-Means algorithm.

	rank	tourney_importance	%ace	%1st_srp_won
rank	1.000000	-0.556388	-0.164412	-0.464810
tourney_importance	-0.556388	1.000000	0.263452	0.192371
%ace	-0.164412	0.263452	1.000000	-0.201896
%1st_srp_won	-0.464810	0.192371	-0.201896	1.000000

Figure 5: Correlation plot of the chosen attributes

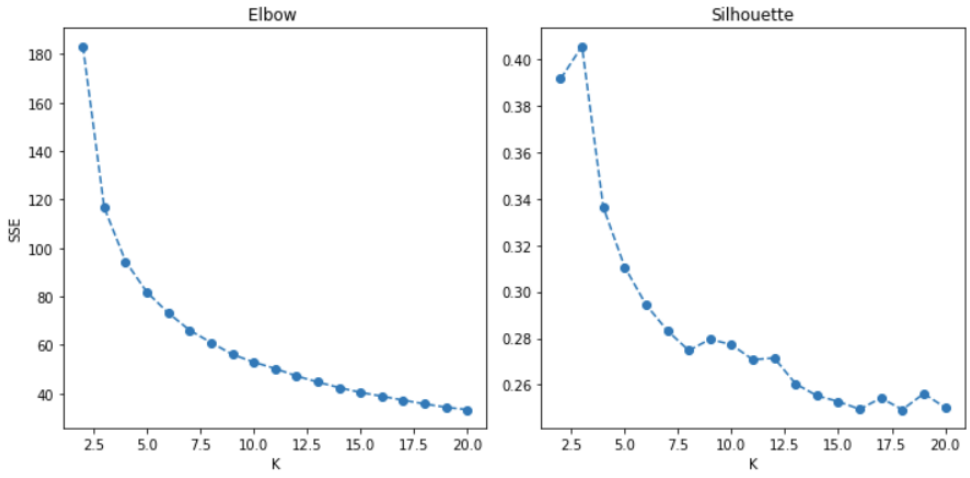


Figure 6: Elbow method with Silhouette

Using the elbow method the best value for k is 3 as it represents a good trade-off between SSE and silhouette score.

Cluster analysis In the plot **Figure 7** the centroids on all the features are reported in order to have a better understanding of the clusters.

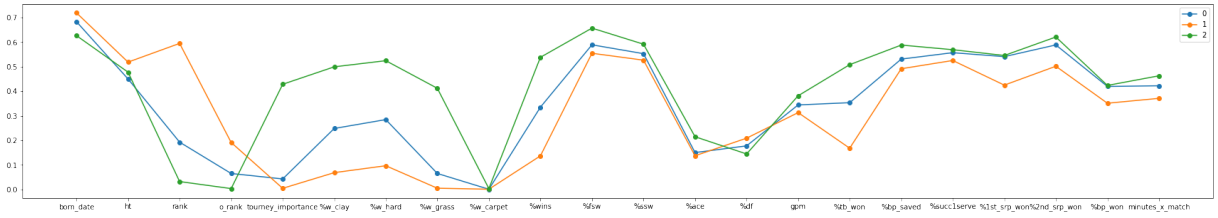


Figure 7: k-means Lineplot

As it can be seen in **Figure 8** the k-means captures three clusters each representing the strength of the players (Strong, Medium, Weak).

- Cluster 2 (Strong players): they are older and possess an average height. They have a high rank and play against equally strong players, in fact, tourney importance is high and the win percentage on all terrains is high as well as the global win percentage. It is important to note the high tie break value indicates a high capability to overcome those situations with a lot of psychological pressure. Regarding the game characteristics, they are really strong in attack as seen in fsw, ssw, ace and have a low percentage of double faults so they make few mistakes during the serve. They have also many excellent characteristics in the defense phase as we can see in %bp_saved, %1st_srp_won, %2nd_srp_won. While concerning the duration of the matches we see how the minutes per match are the highest as well as the games per match to indicate that the matches are played by high-level athletes.

- Cluster 1 (Weak players): they are the youngest and also the highest players, have a very low rank, and play with much stronger opponents than them. The tourney importance is low as well as the global win percentage and the ones for each type of terrain. Regarding the game characteristics, they are weak in both in attack and defense, we see in particular how the percentage of double faults is high and how the percentage of %tb_won is low, indicating young and inexperienced players who cannot manage the decisive part of the game well. Regarding the duration of the match, the minutes per match are really low as well as the games per match, this could be explained by the fact that the match is not fought or they face very strong opponents.

- Cluster 0 (Medium players): they are players with an average age but with a low height, their rank is in the average and they play with players of the same level. Regarding the percentages of victories, they turn out to be exactly between the strong and the weak. We can say that in all the features they are found in the middle especially in %tb_won, gpm, minutes per match. While regarding the characteristics during the game, in the attack phase they are not so strong, in fact, as we can see in the ace percentage, they are very close to the performance of cluster 1, but in the defense phase they are really strong as we can see in succ1srtve, %1st_srp_won, %2nd_srp_won %bp_won they are similar to the strongest if not the same in some places. So this cluster could represent those players with average experience who are strong on defense but need to improve on offense to be complete and therefore be considered strong.

We report information of the cluster found, with a total silhouette score of 0.4058, in **Figure 8** and **Table 1**.

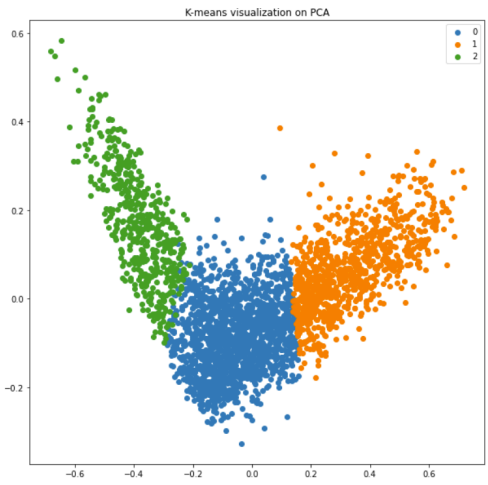


Figure 8: Kmeans on PCA

cluster	size	silhouette	type
0	1651	0.4201	medium
1	800	0.3535	low
2	522	0.4484	hight

Table 1: Clusters size and silhouette

4.4 DB-scan

The dataframe used for DBSCAN is the same used for K-Means, with the same selected subset of attributes. The distance between data points is computed using the Euclidean distance resulting in a square matrix containing the pairwise distance between all pairs of points. Using the Knee method with k from 2 to 20 we can notice in **Figure 9** that the

curve of the distance behaves almost in the same way regardless of the value of K.

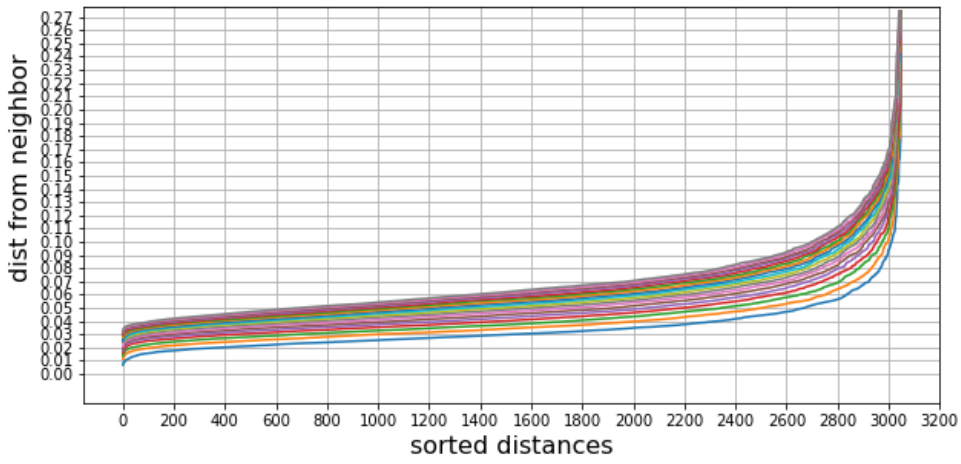


Figure 9: DBSCAN knee

The elbow is localized x=2800 and x=3000 with the corresponding values of eps between 0.07 and 0.14. At this point, the grid search method is exploited to find the best values for the two parameters of the DBSCAN algorithm: epsilon and min-samples. To optimize the search:

- it is applied only on those values of eps that are within the windows
- Since for min-samples the grid search is restricted between 2 and 15.

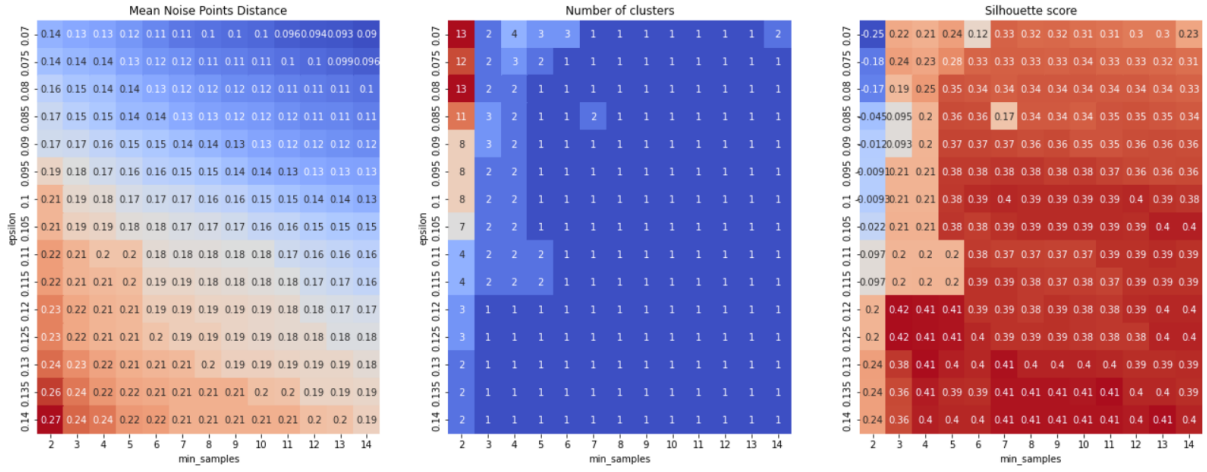


Figure 10: DBSCAN heatmaps of hyperparameters

Using the heatmaps in **Figure 10** we selected eps=0.12 and min-samples=3 as best parameters, resulting in 1 cluster plus the one for the outliers distributed as follows:

Cluster labels	-1, 0
Elements per cluster	14, 3036
Silhouette score	0.4156

DBSCAN cluster understanding The result obtained from DBSCAN applied to this dataset is very unbalanced, and different choices of the parameters do not improve the situation. The cluster labeled as -1 contains data points that are far from the dense area (so distant from the clusters). In the **Figure 11** we can see that these noise points in space do not have an apparent relationship between them, this confirms the fact that they are outliers and they differ from each other by being outliers for one or more specific features (for example rank 1, percentage of ace 0, etc...). In conclusion, the algorithm works by finding the clusters based on the areas with the highest density. In our dataset this approach is not recommended as we have a single dense area of points and therefore the algorithm will tend to have only one cluster.

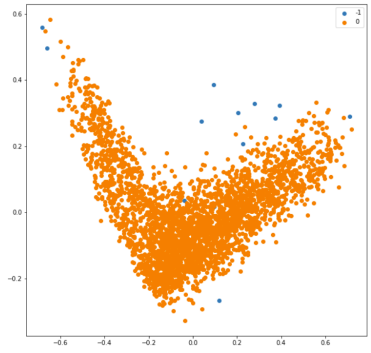


Figure 11: DBSCAN on PCA

4.5 Hierarchical

For the Hierarchical Clustering, to get results comparable in terms of indicators and properties among the clusters, the same set of attributes of the previous algorithms was used. The analysis goes through different methods like complete, single, average, and ward, each of them changes how the distance between two clusters is calculated. The first three are based on mathematical functions while the last one has a more statistical approach. Through the study of the dendrograms, the best value for the threshold was to get a balanced clustering result. In order to evaluate the goodness of the clustering the silhouette coefficient was used.

Method	Cluster	Silhouette	Threshold
Single	3049, 1	0.55	0.25
Complete	2625, 425	0.38	1.20
Average	3045, 4, 1	0.47	0.57
Ward	1780, 1270	0.38	10.50

Even if the silhouette is pretty good, the results for single are bad since the clusters are unbalanced, collapsing all the elements in one single cluster. Best results are achieved with ward method both from the point of view of the number of elements in the clusters and coherence of values among clusters too.

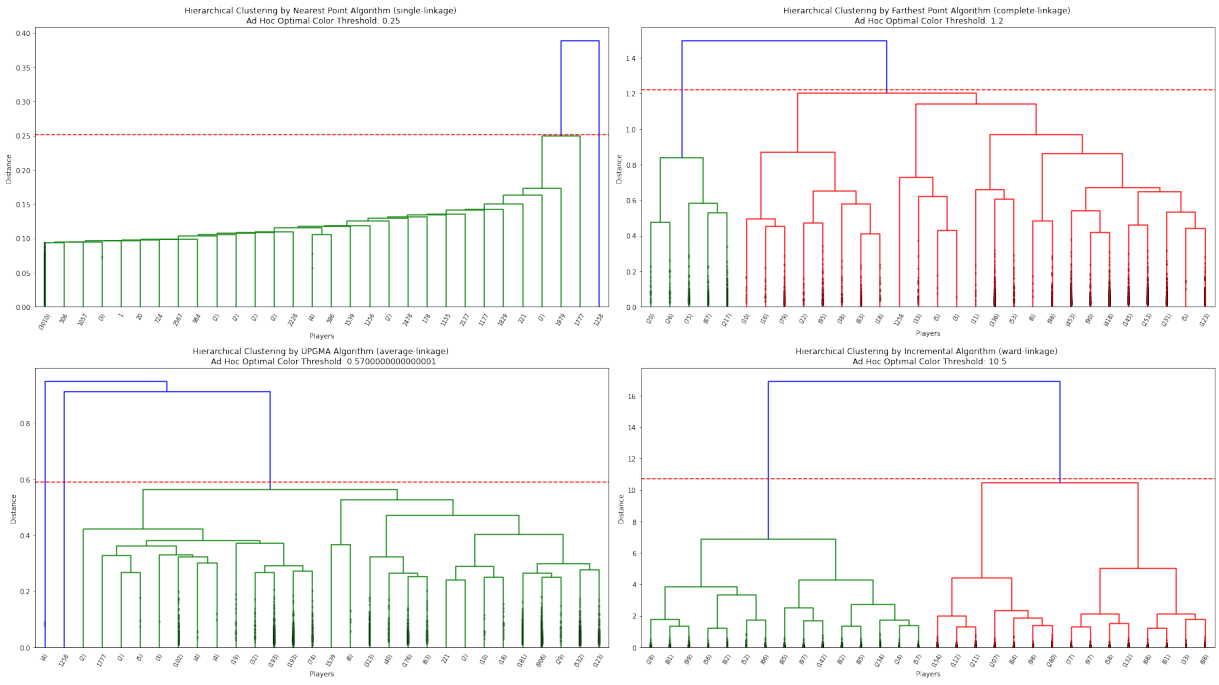


Figure 12: Hierarchical methods

Comparison among the different linkages and dendrograms with ad hoc threshold From the dendrograms and the result previously shown, the first three methods are not able to create a good and balanced clustering, all the items are collapsed in a singular big cluster. With Ward's method, instead, the result is more balanced, and it is possible to study the characterization of the elements inside the two main clusters. The results are similar to the ones obtained from K-Means in which the clusters seem to represent very

strong players and weak ones.

4.6 Alternative clustering techniques

For the clustering analysis also tried alternative clustering techniques like X-means and Fuzzy-k-means.

4.6.1 X-means

X-means is a k-means based algorithm that solves its two main problems: finding the best value of k and the computational cost. At each iteration, the algorithm applies k-means and subsequently performs the cluster splitting which optimizes the value of K by deciding which subset of the current centroids must be divided in such a way as to have a better fit. The splitting decision is done by computing the Bayesian Information Criterion (BIC). In this way, it does not look for the value of k but it will automatically be based on the dataset.

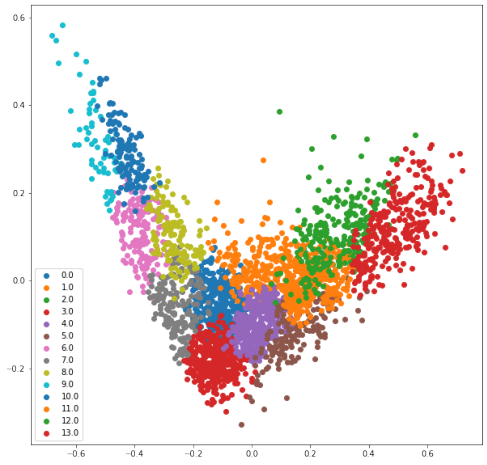


Figure 13: X-means clustering on PCA

Cluster labels	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
Elements per cluster	356, 222, 131, 386, 372, 162, 173, 185, 140, 60, 124, 316, 133, 290
Silhouette score	0.254

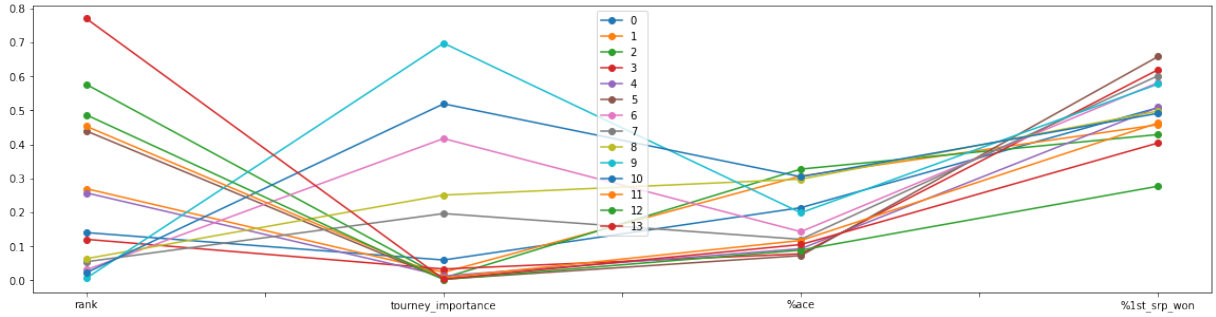


Figure 14: x-means lineplot

The results obtained with x-means are totally different from the previous approaches, in this case, the algorithm managed to find 14 clusters with a silhouette score of 0.25. Through the lineplot, we can see that each feature is layered on the clusters. In this way, having more groups we can see the internal differences between the strongest, average and poor players who differ in some specific features.

4.6.2 Fuzzy-c-means

Fuzzy-c-means is a clustering algorithm based on fuzzy logic. In particular, a point is not forced to belong to a cluster or to another because through the fuzzy logic, data points can potentially belong to multiple clusters with a value that goes from 0 to 1. The fuzzy c-means algorithm is very similar to the k-means algorithm and it differs by the membership values in the objective function and the fuzzifier value m. The fuzzifier m determines the level of cluster fuzziness. A large m results in smaller membership values and hence, fuzzier clusters. When m is set to 1 the membership converges to 0 o 1 resulting in a crisp partitioning. In absence of knowledge of the domain, the default value is set to 2. The algorithm minimizes intra-cluster variance as well, but has the same problems as 'k'-means; the minimum is a local minimum and

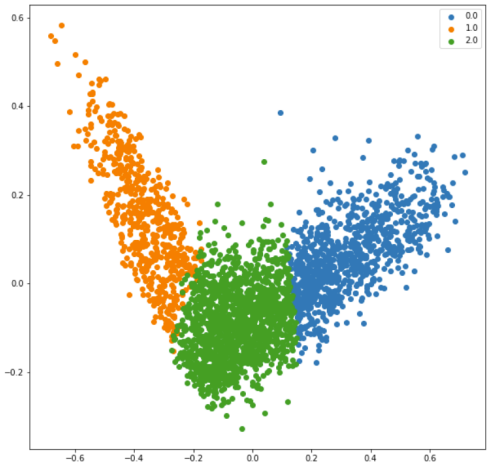


Figure 15: Fuzzy-K-means on PCA

the results depend on the initial choice of weights. We use the same hyper-parameter K=3 found in the k-means approach in order to compare the two algorithms and we have obtained these results:

Cluster labels	0, 1, 2
Elements per cluster	906, 596, 1548
Silhouette score	0.39

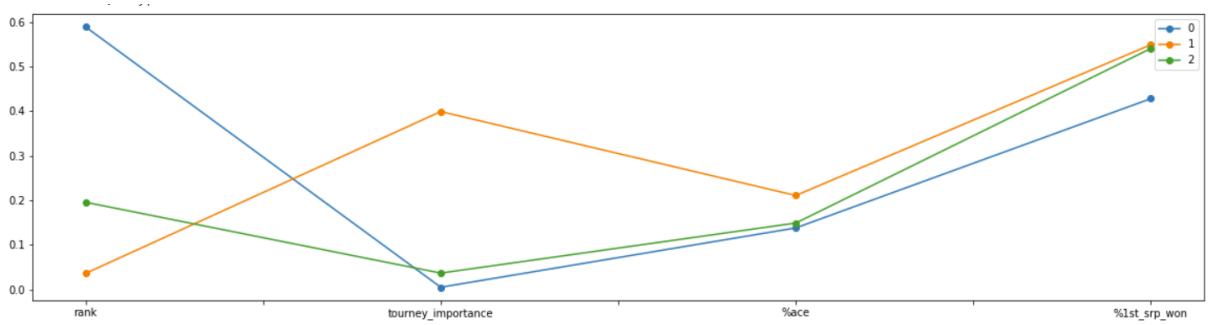


Figure 16: Fuzzy-k-means lineplot

The results obtained are in line with our estimate, the two algorithms provide similar results. In fact, we can see in the line plot that the three clusters represent strong players, medium players, and weak players. The main difference we can see is in the silhouette score due to the different sizes of the three clusters. This is caused by the fact that through fuzziness we can have a different distribution of data points where a point can belong to more clusters but, in the end, each cluster contains points that most likely (in line with membership) belong to that cluster.

5 Classification

The aim of the classification task is to predict for each player a label that defines if s(he) is a high-ranked player or a low-ranked player (binary task) by exploiting the feature related to the rank of the players. At first, we have to prepare the dataset for the labeling:

- Remove record with missing rank value
- Drop the height column since we have too few records.

- Drop IOC and o_rank since we don't want a prediction that uses these features.

Using the results given by the clustering task we extracted the labels, using the value 436 of the rank, which is the middle cluster, as the threshold to discretize it. After the labeling we have a balanced dataset with 1412 High and 1638 Low. Before starting the predictive analysis the dataset has been modified:

- Discretization of the categorical features for the models that don't depend on the distance of the values
- Hot Encoding of the categorical for the other models.

Then we split the dataset using 30% as test set and fitted the models on the remaining performing a grid search where possible. Furthermore, for NN, KNN, and SVC, the dataset has been scaled with a MinMax scaler.

5.1 Results

We report the results over all the models highlighting the best 3 based on the validation accuracy, moreover also a comparison with the roc curve, reported in **Figure 17**, has been done.

Model	TR Accuracy	VL Accuracy	TS Accuracy
Decision Tree	0.92	0.91	0.90
Gaussian Naive Bayes	0.86	0.86	0.86
Multinomial Naive Bayes	0.86	0.86	0.85
Random Forest	0.94	0.91	0.91
AdaBoost	0.92	0.91	0.91
Neural Network	0.93	0.88	0.91
KNN	0.88	0.86	0.85
SVM	0.92	0.89	0.90
Rule Based	0.89	0.89	0.88

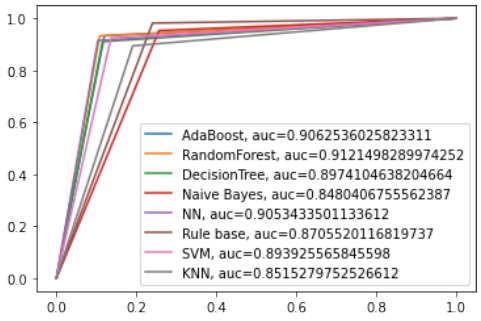


Figure 17: Roc curve on Test

As we can see from the results there is a little difference between the decision tree and the ensembling methods, this because probably in our dataset we can't achieve a lot more than 0.91 so the DT already is able to classify quite well as an ensembling method. This results confirm what we have studied, namely that for tabular data the decision tree is the state of the art and the ensembling methods performs better than the singular.

Random Forest For brevity we show only the results of the random forest over the test set. True labels can be seen in **Figure 19a**, instead, in **Figure 19b** are highlighted the misclassified samples using %ace and tourney_importance to plot the scatter, furthermore also the confusion matrix is reported in **Figure 18**. We can clearly see from the confusion matrix that we have a good balance between precision and recall.

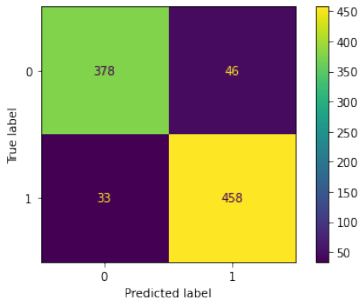
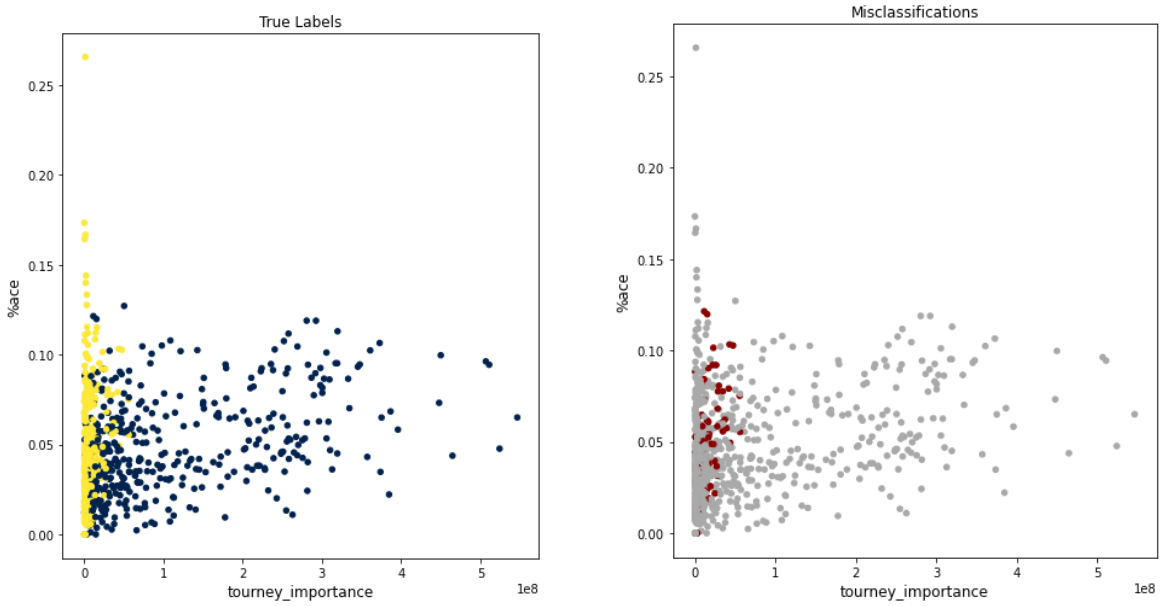


Figure 18: Confusion matrix on test set



(a) True labels

(b) Misclassified points

Figure 19: Misclassified points compared with true labels

6 Explanation Analysis

The explanation is done on SVM, and Random Forest, for the sake of brevity we report only the result on the Random Forest. Furthermore, we compare results over 2 different types of RF, one using tourney_importance and one not, this because in the explanation it has a high contribution but we think that it is not so useful if one tries to understand

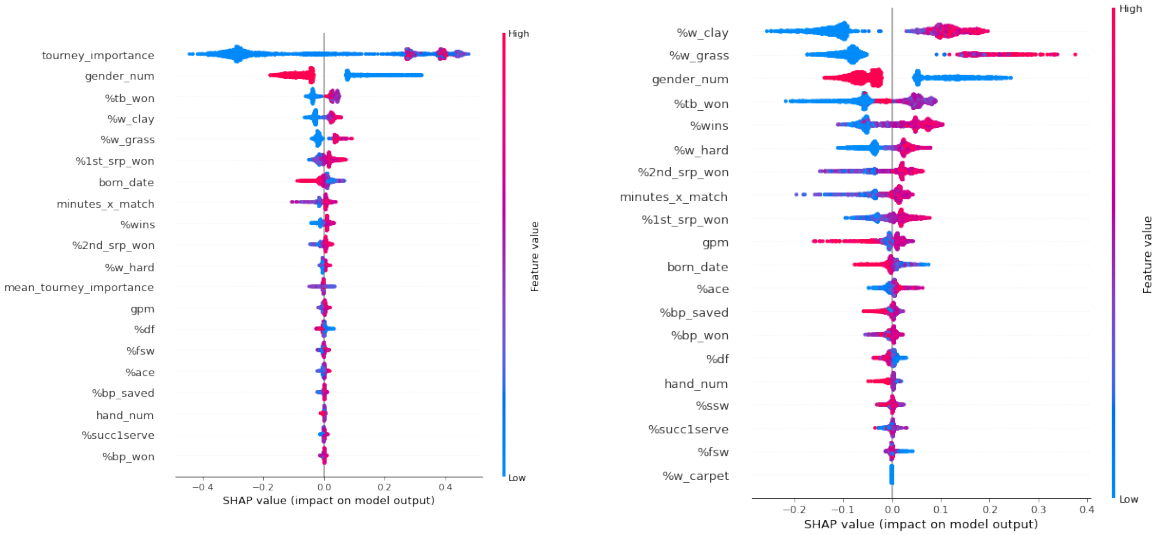
what a player needs to improve.

6.1 SHAP

SHAP is a method to explain predictions, based on the game theoretically optimal Shapley values. SHAP can provide local and global explanations. From the global one, we can see that the tourney_importance is the most important feature for both labels 0 and 1. The SHAP value plot can further show the positive and negative relationships of the predictors with the target variable. This plot is made of all the dots in the train data. It demonstrates the following information:

- Feature importance: Variables are ranked in descending order.
- Impact: The horizontal location shows the contribution to the output.
- Feature value: Color shows whether that variable is high or low for that observation.

We can see in **Figure 23b** the top 20 features:



(a) SHAP with tourney importance

(b) SHAP w/o tourney importance

Figure 20: Global explanations with SHAP

Removing tourney_importance and mean_tourney_importance we can see that the explanation is better from a player statistics point of view. We can also see in both cases that gender is an important feature. We also tested SHAP on local explanations with

both models. The results are aligned with the global explanation in which without the tourney_importance feature we gain in the explanation at the cost of the fidelity.

6.2 LIME

Lime is a model-agnostic approach that attempts to understand the model by perturbing the input of data samples and understanding how the predictions change. Lime only approximates the black-box locally, where the task is simpler. We can see that the prediction probability is higher with tourney_importance as we have seen with shap, also confirming the gain in the explanation.

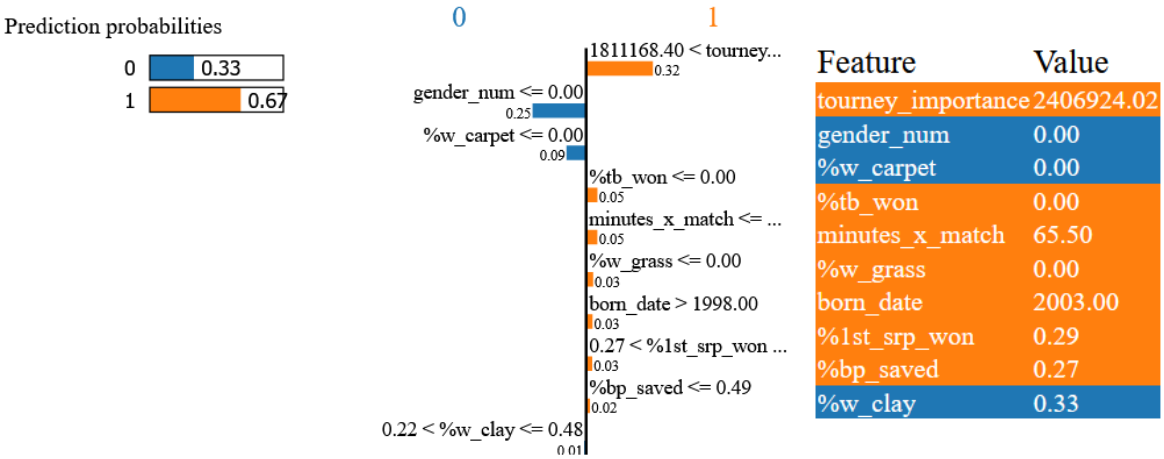


Figure 21: LIME Explanation of Weak player prediction using tourney_importance

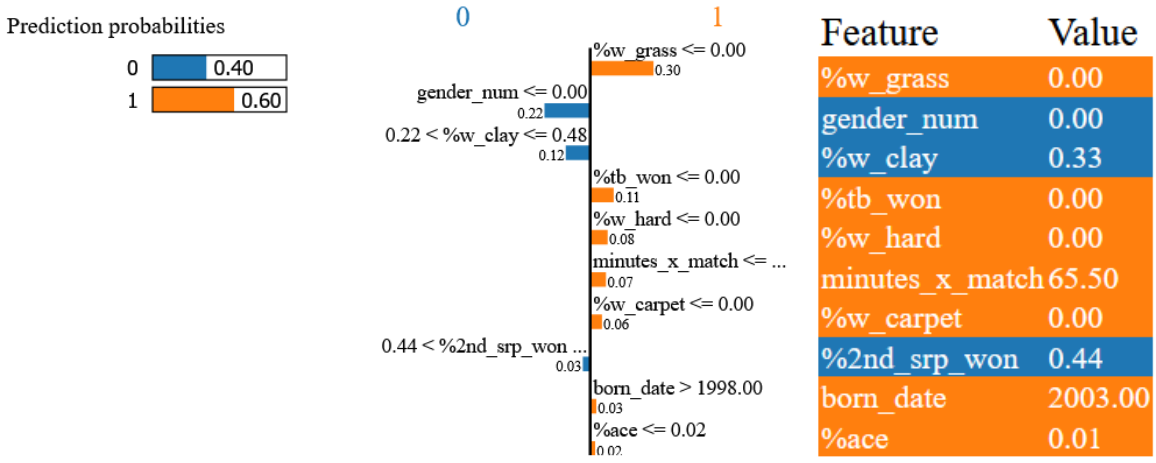
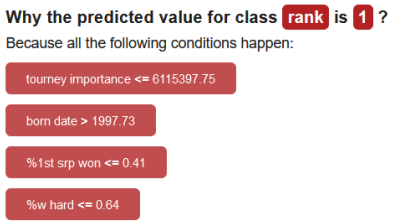


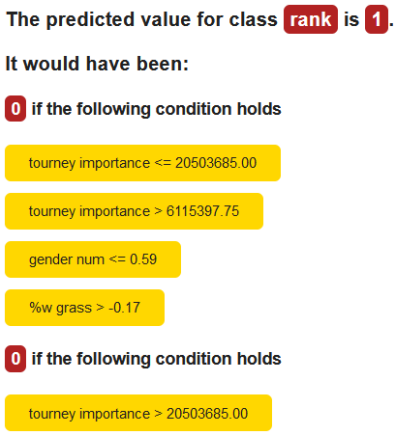
Figure 22: LIME Explanation of Weak player prediction w/o tourney_importance

6.3 LORE

LORE first learns a local interpretable predictor on a synthetic neigh- borhood generated by a genetic algorithm. Then it derives from the logic of the local interpretable predictor a meaningful explanation consisting of a decision rule, which explains the reasons of the decision; and a set of counterfactual rules, suggesting the changes in the instance's features that lead to a different outcome. In **Figure 24** we can see the proposed explanation and the counterfactual. They are mainly based on the tourney importance feature and do not provide useful information for an athlete who wants to improve his skills.

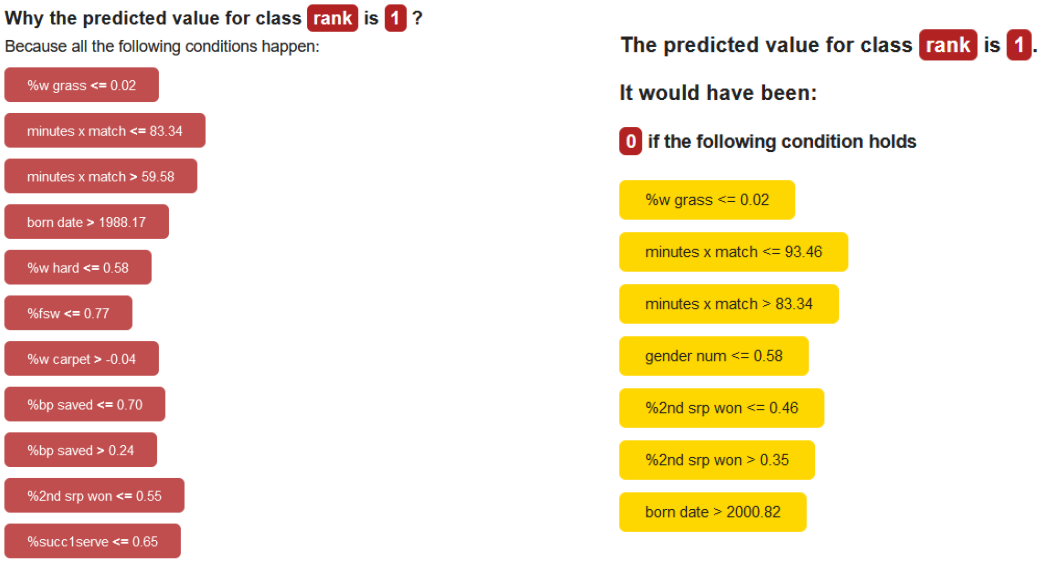


(a) Explanation



(b) Counterfactual

Figure 23: LORE Explanation of Weak player prediction using tourney_importance



(a) Explanation

(b) Counterfactual

Figure 24: LORE Explanation for the predicted class Weak player without tourney_importance

While applying LORE to the trained model without tourney_importance this provides us more informative explanations and counterfactual, in particular in this way it is possible to understand what a player needs to improve to become stronger.

6.4 Evaluation

In the context of feature importance based explanations, we can evaluate the explanations by using:

- Faithfulness: reveals the correlation between the importance assigned by the interpretability algorithm and the effect of each of the attributes on the performance of the predictive model.
- Monotonicity: evaluates the effect of the explanation by querying the black-box model, by incrementally adding each attribute in order of increasing importance.
- Fidelity: evaluates how good is the surrogate model at mimicking the black-box decisions. Available only for methods that construct a surrogate model.

Since SHAP doesn't use a surrogate model, to compare it with LIME we only use faithfulness and monotonicity.

Model	Faithfulness Mean	Faithfulness std	Monotonicity
SHAP with tourney	0.475	0.442	False
LIME with tourney	0.555	0.417	False
SHAP without tourney	0.477	0.261	False
LIME without tourney	0.398	0.202	False

Table 2: Explainability table evaluation

The reason SHAP remains with a similar faithfulness after removing tourney_importance could be that shapley values are calculated by removing the effect of specific features which is similar to how faithfulness is computed. The two methods have both false monotonicity, the reason could be that LIME trains a local surrogate model to imitate the behavior of the original one, but does not guarantee the same behavior globally, and since permutation randomly shuffles the features, this does not guarantee monotonicity. As for SHAP, it is calculated by removing features rather than adding features. In conclusion, SHAP seems to maintain a good faithfulness in both cases, thus more able to understand which are the important features.

7 Conclusion

For the analysis of this tennis matches data set, we used different data mining techniques, trying to understand which are the best features to create a player profile. In the Data Understanding section, we addressed issues related to data semantics and quality, like fixing missing values, outliers, and removing redundant variables. Our main goals were to understand the features, clean up and reduce the dimensionality of the dataset. In the Clustering section, we concentrated our focus on numeric attributes to find groups of similar points in the data set. We found that the K-MEANS algorithm better captures the cluster in the point clouds in **Figure 7** with respect to the other algorithms. In the classification section, we tried different models and it turns out that the best are ensemble and Decision trees that confirm why they are the state of the art. After the classification we provide explanations of our black boxes to see which features are more relevant and

how they behave in local and global explanations.