LUISS

LUISS Guido Carli

Department of Business and Management

Bachelor's Degree in Management and Computer Science

Course of Data Analysis for Business

Data-Driven Approaches for Soccer Match Analysis

Supervisor:

Prof. Francesco Iafrate

Candidate:

Martino Olivieri

248021

Accademic Year 2021/22

Summary

This thesis explores possible solutions to automate match analysis in soccer. It makes use of the spatial-temporal data set published by the *Wyscout* company. The first part will be entirely dedicated to exploring and visualizing the data together with a concrete description of possible applications. While the second part is completely devoted to the definition and implementation of two machine learning models: the first is supervised and is used to get a numerical evaluation of every player. In contrast, the second is a clustering model that detects relevant areas on a soccer pitch. The analysis was entirely done by exploiting the possibilities offered by the R programming language, while the parsing of the JSON files was done using Python; details regarding the script can be found in the appendix.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to match analysis powered by data

As every team sport, soccer is becoming more and more based on strategies. The Oxford Dictionary defines tactics as "a carefully prepared action or strategy to achieve a specified aim." Naturally, the goal of competitive soccer is to win the game. For this reason, the coach's role in a soccer team is evolving, requiring many more skills with respect to the past. Hence new figures, such as a team of match analysts, are now starting to complement the coach. Data is also central in modern soccer, as many decisions are powered by analytics. A quick example could be predicting a player's injuries based on physical and spatial information [1]. These kinds of applications have a significant impact even on the financial part of a soccer company since an injury could cause substantial monetary losses. Data science in soccer is applied mainly in match analysis which is the objective study of all tactical and technical aspects that a soccer player or team performs during a match. The main issue regards collecting valuable data since it requires advanced hardware and software tools that only a few companies can afford. *A.S Roma* is one of them and Its coach, Jose Mourinho, uses spatial-temporal data during and before the match for decision-making. In the past, the analysis of a match was performed by experts. After looking at all the opponent's previous games, they would compile a report containing the adversary's strengths and weaknesses. Nowadays, by exploiting new technologies, some analysis processes could be automated and hidden patterns could be discovered.

1.2 Collecting data in soccer

The previous section highlighted how difficult and expensive it is to collect spatial-temporal data in soccer. "Aside from occasional attempts, soccer statistics have only recently been produced, thanks to sensor technologies that offer high-fidelity data streams taken from every match" [4]. There are three main methods:

- **GPS**: The soccer companies can track their players via *GPS* devices during both games and training.
- **Computer Vision**: The data can be collected using object tracking models.
- **Soccer-log**: Description of the events that occur during a match collected through proprietary tagging software.

The first two are automated and are considerably more complete and precise, while the third is semi-automated. The main difference between the two is that one provides continuous data (both offensive and defensive phases), while the second only describes events regarding the ball possession. Due to the scarce data availability of GPS and Tracking data, this thesis explores the possible application of the soccer-log data-set published by *wyscout*.

1.3 Introduction to the structure of the thesis

This thesis is structured in two main sections that are actually connected to each other. The first phase of the pipeline regards the exploration and visualization of the data together with a concrete description of possible applications for a coach. This section will try to discover patterns in soccer games: starting from the big picture on the whole dataset and then going deep into the performance of the single teams and players. Instead, the second phase is entirely dedicated to machine learning. Both supervised and unsupervised methods will be applied to discover hidden patterns.

Chapter 2

Data Pre-Processing

2.1 Data description

As stated in the last section, the most challenging part is Collecting the data. The Italian Company *Wyscount* published a free-to-use database that comprehends the season 2017/2018 of the top 5 European national soccer competitions: Spanish first division, Italian first division, English first division, German first division, and French first division. The UEFA nation coefficient indicates that these competitions are the most significant in Europe [4].A total of seven data sets include details on all tournaments, games, teams, players, events, referees, and coaches. The JSON format (JavaScript Object Notation) is used to convey each data set. For the analysis, only the following tables will be used:

Events is the main table used because it describes every event in every match. The following are the Information contained:

- **eventId**: The identifier of the event type;
- **EventName**: The name of the event;
- **subEventId**: The identifier of the sub event;
- **subEventName**: The name of the sub event;
- **tags**: A list of tags related to the event;
- **eventSec**: The match' second during which the event took place;
- **id**: The unique identifier of the event;

- **matchId**: The identifier related to the match to which the event refers;
- **matchPeriod**: In which period the event took place. First or second half;
- **playerId**: The unique identifier of the player that generated the event;
- **positions**: The coordinates of the pitch where the event started and ended;
- **teamId**: The unique identifier of the event to which the player belongs;

Matches contains information related to every match present in the dataset. The following are the columns:

- **competitionId**: Refers to the unique identifier of the competition;
- **date**: The date and time when the match started in implicit form;
- **duration**: The duration of the match. It can be: "Regular", "Extra Time" or "Penalties";
- **gameweek**: The number of week from the start of the league;
- **label**: The result of the match in the following form: "home Team - Away Team, score";
- **seasonId**: Refers to the season of the match;
- **status**: If the match has been already played or not;
- **venue**: The stadium in which the match was played;
- **winner**: The unique identifier of the team that won the game;
- **wyId**: The unique identifier of the match;
- **teamsData**: The data refereed to both the teams. It contains information about the formations, the bench, substitutions, coaches and score;
  - **hasFormation**: Binary variable: 1 if a formation is present, 0 if not;
  - **score**: Number of goals scored during the match not including penalties;

- **scoreET**: Number of goals scored during the whole match including extra time (not counting penalties);
- **scoreHT**: Number of goals scored during the first half of the match (not counting penalties);
- **side**: Weather the team was home or away;
- **teamId**: The identifier of the team;
- **coachId**: The identifier of the coach;
- **bench**: the list of player present in the bench, together with some basic statistic for every player related to the match;
- **lineup**: The list of players in the line-up together with come statistics related to the match for each player;
- **substitutions**: The list of substitution that took place during the match;

Players contains all the information related to every player present in the data set. The following are the Information contained:

- **birthArea**: Information about the birth area;
- **birthDate**: The birth date of the palyer (YYYY-MM-DD);
- **currentNationalTeamId**: The unique identifier of the national team in which he plays;
- **currentTeamId**: The unique identifier of the team in which he plays;
- **firstName**: The first name of the player;
- **lastName**: The last name of the player;
- **foot**: The preferred foot of the player;
- **height**: He height of the player (in centimeters);
- **middleName**: The middle name (if any) of the player;
- **passportArea**: The geographic area associated with the player's current passport;
- **role**: The main role of the player;

- **weight**: The weight of the player (in kilograms);
- **wyId**: The identifier of the player, assigned by Wyscout;

Tag contains every possible tag present in the ”events” data table to describe each observation: the following are the features:

- **Tag**: The unique identifier of the tag;
- **Label**: The actual name of the tag;
- **Description**: The description of the tag;

2.2 Exploratory data analysis

The first step is the *exploratory data analysis (EDA)*. Usually it is the first phase toward a good report. It comprehends all the methods to investigate relationships between variables and provides an overall description of the tables mainly through data visualization. The EDA process will be focused on the *events* table, since it will be the only one used in the modelling section. The analysis could already result in insightful information for a team's coach. So, together with statistical reports, a possible application will be described.

Event Name

The first interesting variable to visualize is *EventName* which contains what event took place from a set of 10 possible levels.

Fig.2.1 shows that *pass* is the most frequent observation. This was an expected result for who knows a bit about soccer, but it is always good to have statistical confirmation.

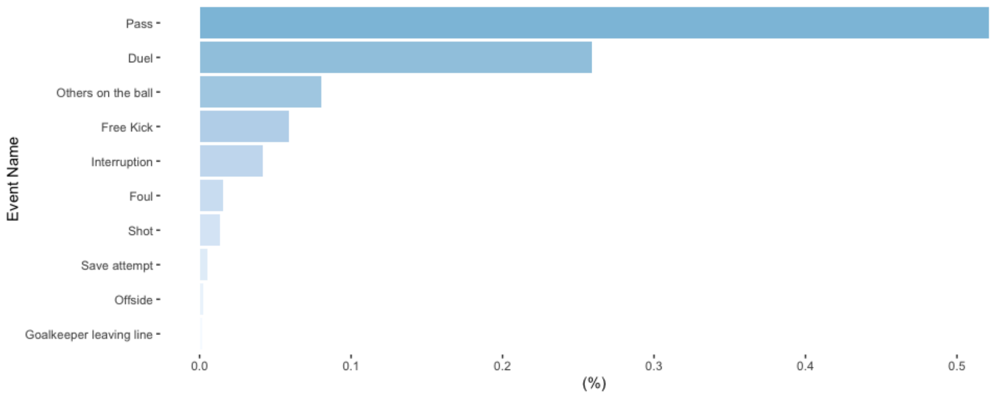

Figure 2.1: Frequency of each event type

Instead the Fig.2.2 shows the heat map of the 4 main events. The *pass* event is very dense in the central area of the football field (fig. a), while the *shots* are mostly concentrated in the penalty area (fig b). For the *Duel* and *Fouls* there is no evident pattern.

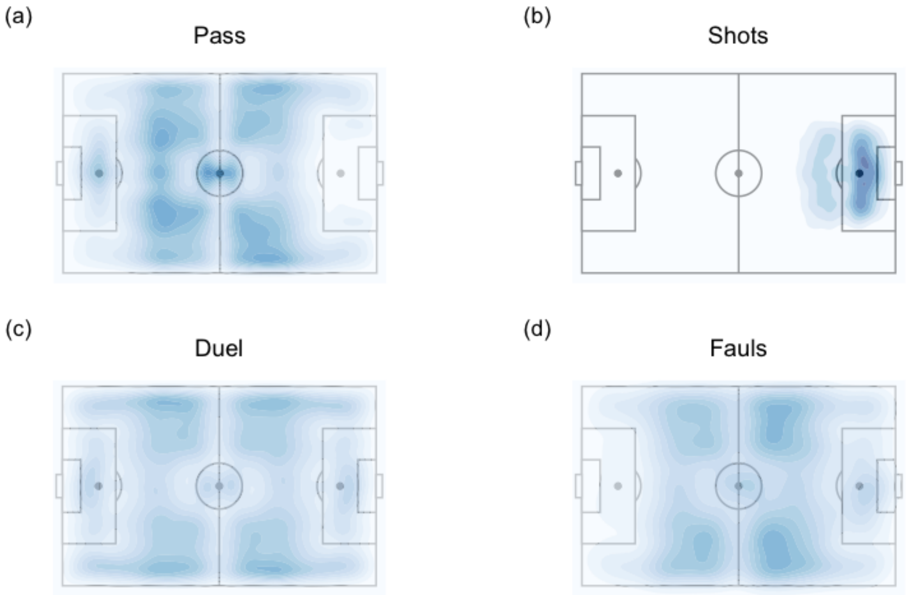

Figure 2.2: Heat map of the most relevant events type

Goal and Pass events

Fig.2.3 shows the statistics of the *goal* and *pass* variables. In particular, fig (a) and (b) demonstrate that in many matches around three goals are scored while in only a few of them more than 4. Instead, fig (c) and (d) shows the density of respectively accurate and not accurate passes. The mean in each game is 740 for precise passes and 148 for the not accurate pass. Furthermore, it is clear that the two distributions are normal, but on average, more accurate passes are completed with respect to non-accurate. x

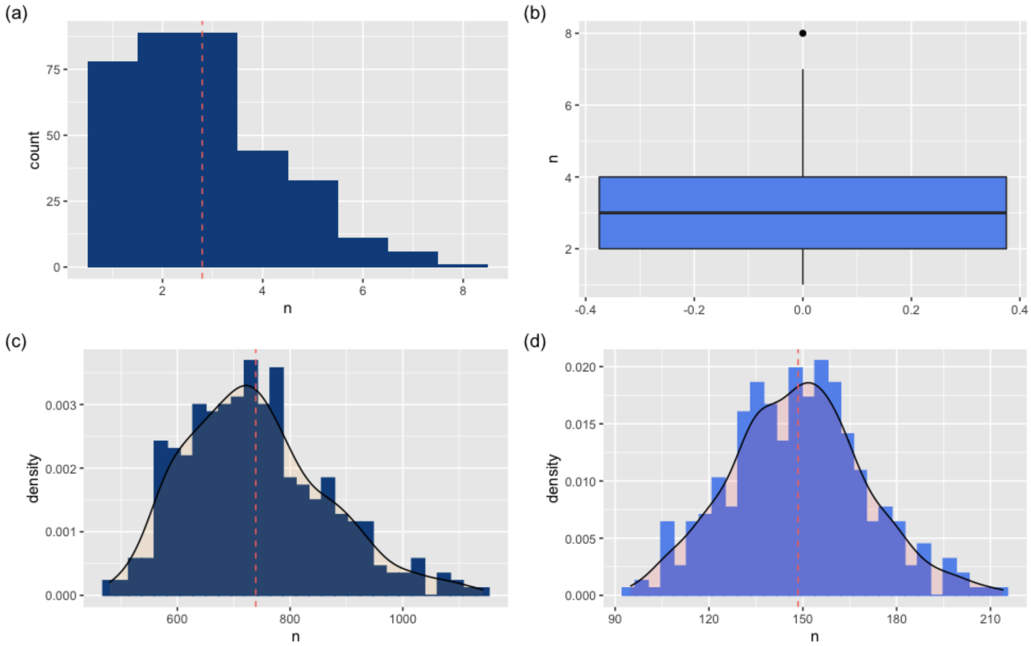

Figure 2.3: Distributions of the *goal* and *pass* column

Distribution along the 90 minutes

The fact that the data are spatial-temporal is essential for the analysis. We could use the *eventMin* column, not just in the ML model, but also to get insightful statistics and visualization. For example, Fig. 2.4(1) shows the frequency of the red cards in each time frame of the matches. The referees seem to give more red cards at the end of the game, maybe because the players are tired or stressed by the result. Instead, Fig.2.4(2) represents the frequency of the yellow cards in each time frame: apart from the start of the game, during which the referees tend to give fewer yellow cards, the frequency looks flat. At the same time, Fig.2.4 (3) regards the frequency of the goals that are well distributed along the 90 minutes.

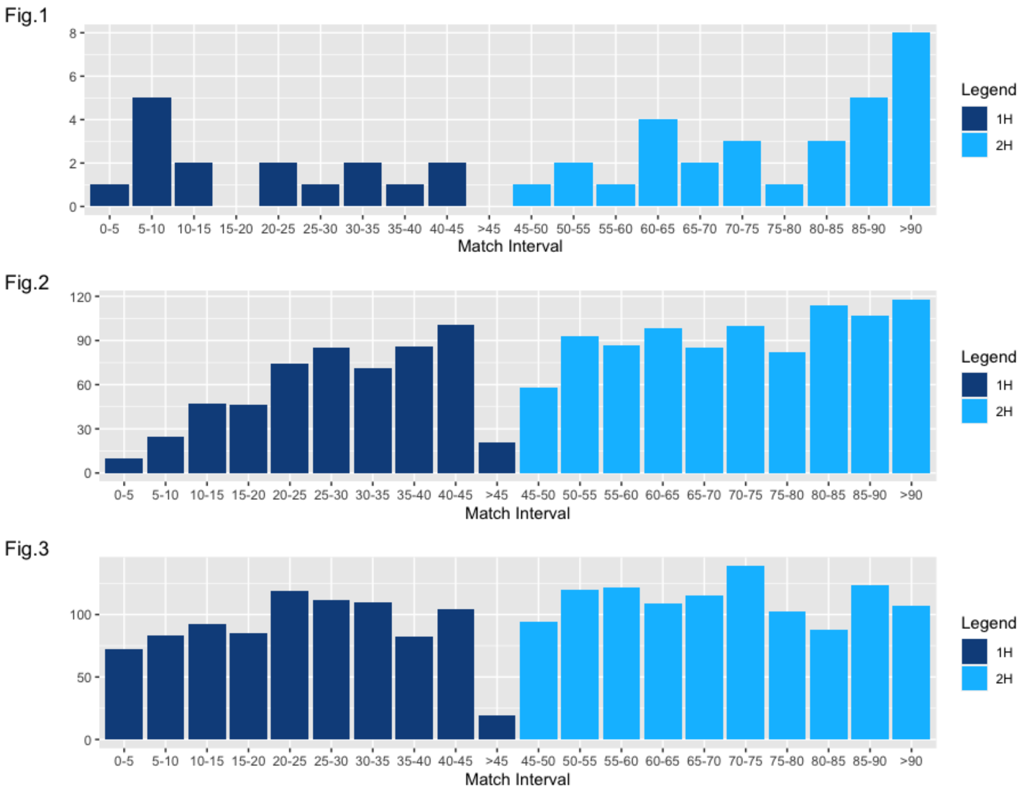

Figure 2.4: Frequency of the red cards in each time frame of the matches

Focus on *Roma - Chievo Verona 4-1 2017/18*

We have a structured agglomerate of data from the season 2017/18. So it would be good to use visualization to understand the whole data set as shown in the last sections. Still, it would be even more insightful to analyze one match in particular from a statistical point of view. For this reason, the match *Roma - Chievo Verona 4-1 2017/18* was selected and analyzed.

Firstly since we have the spatial data about every event, it could be informative to visualize where the goals were scored. Fig.2.5 shows the position in which the players scored a goal during the match. The same visualizations is shown in Fig.2.6, but here the data points represent the shots.

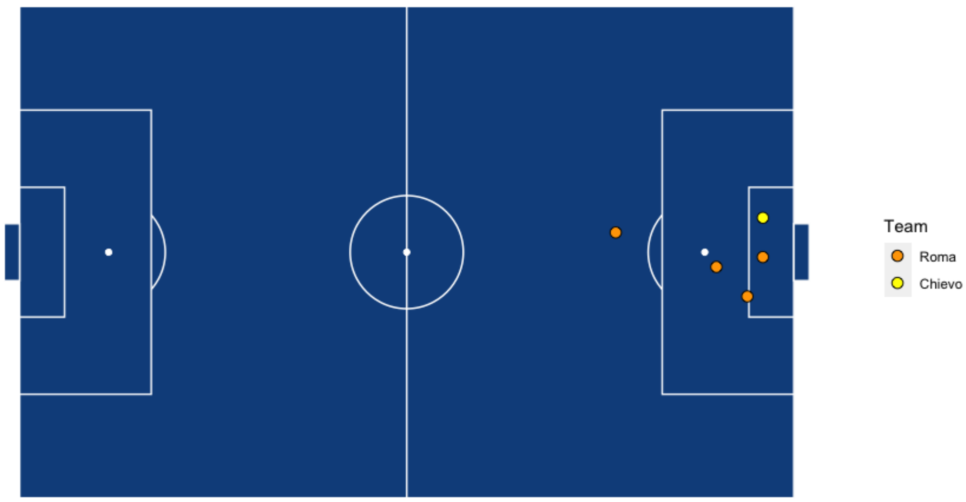

Figure 2.5: Position of the goals scored in *Roma - Chievo (4-1)*

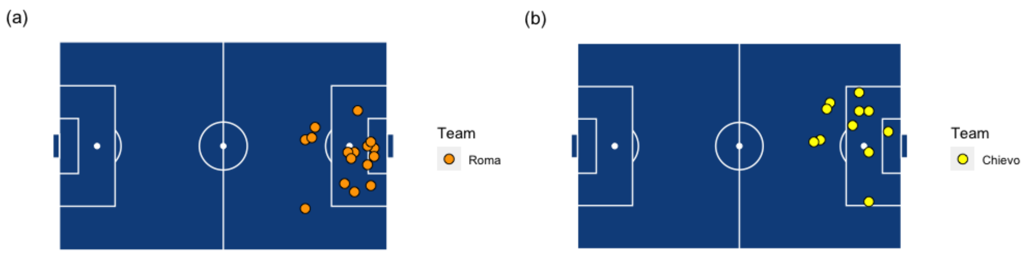

Figure 2.6: Shots from both teams in Roma - Chievo (4-1)

The same analysis could be enlarged on the whole season of a particular team. In this case, using the *wyscout* dataset, fig Fig.2.7 shows a possible visualization of both *Roma* and *Chievo*. In particular fig (a) shows where Chievo Verona scored most of the goals. Fig. (c) and (d) show in which area of the field the teams completed more passes. Instead fig (b) shows where the right back, *Aleksandar Kolarov* played most of the season. All this information together would be very precious to build a strategy for a specific game.

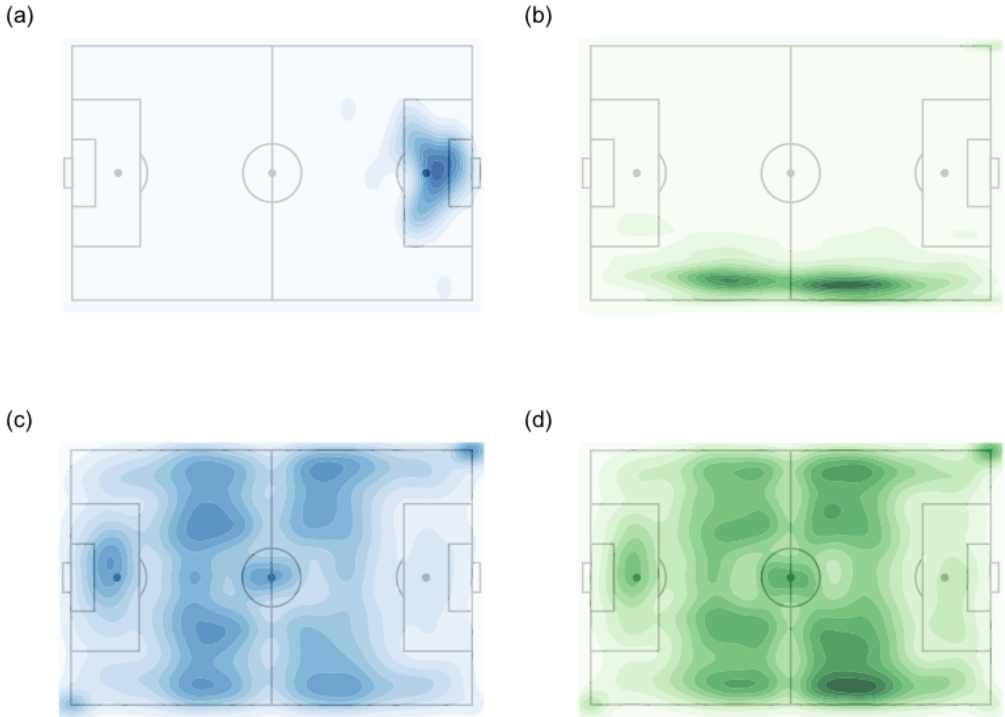

Figure 2.7: Possible report of Roma - Chievo (4-1)

Network Analysis

A more sophisticated visualization concern the analysis of the network of a team. The data structure is not ready for this plot, so some transformation must be applied. To build a network of passes between players of the same team, there should be a sender(who pass the ball) and a receiver(who receives the ball), and both the data should be on the same row. To do so, the following steps are implemented:

- *Apply filter to retrieve only the pass observations;*
- *Place a mark where the ball possession change;*
- *If (the ball possession does not change):*

  - *Select idPlayer in row(i+1);*
  - *Add it to row(i)*

- *Group by sender and receiver*

The resulting table will be composed by three columns (receiver, sender, weight) which are what a network asks as input. Fig. 2.8 shows the graph of *A.S. Roma* in the game *Roma - Chievo Verona 4-1 2017/18* with the following components:

- **Nodes:** Players
- **Edges:** Pass completed
- **Weights:** n of passes completed

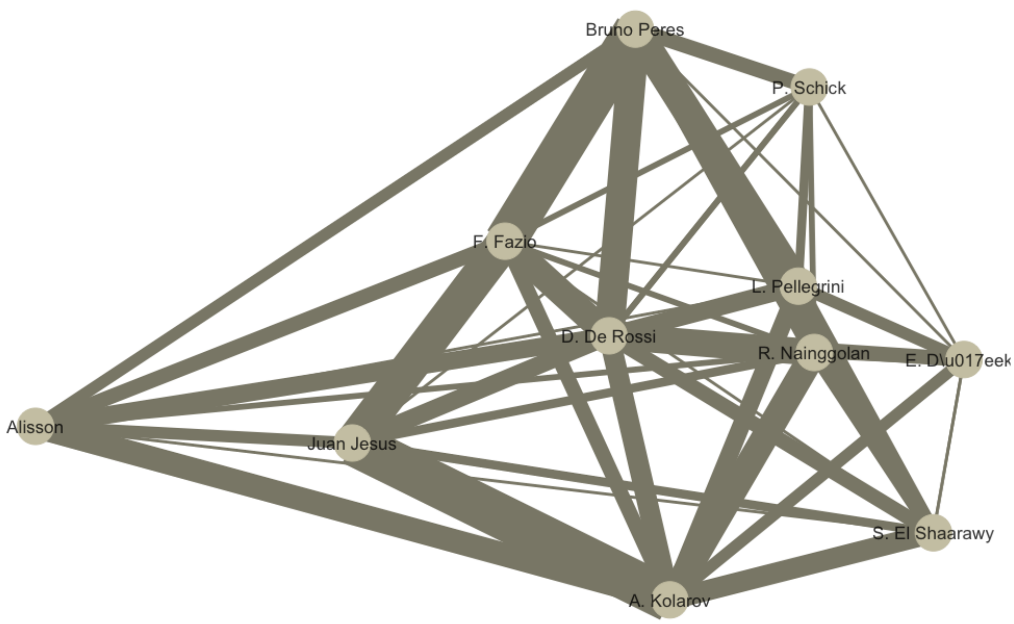

Figure 2.8: Network of passes for A.S Roma in *Roma - Chievo Verona 4-1 2017/18*

For a coach, a set of networks of the last matches played by the team he will face next is fundamental. Maybe a strategy could be to place a player between two highly connected nodes in order to break their ball possession. In this particular case for example, the defenders seams to be highly connected.

Playing with two strikers that press them could be a winning strategy. There exists several possible application of the analysis of a soccer team network. In the paper *Using Network Science to Analyse Football Passing Networks: Dynamics, Space, Time, and the Multi layer Nature of the Game* [5], the authors try to analyze how the network a team changes during a game.

2.3 Data cleaning

Data cleaning is often the most time-consuming part of the process. Data must be processed to be ready to use in the modeling part. Otherwise, the fitting could incur in some problems. The analysis will be focused on the "events" table, but additional features will be taken from the other data-sets. To begin, the first thing to do is to exclude all the observations and the columns that will not be used to make later transformations lighter. For this reason, a good understanding of the topic is needed. To make a quick example, some of outliers should be considered not just from a statistical perspective but also in a concrete sense. Observations such as side fouls should be deleted to make a good clustering model. Firstly only the following observations from the *tag* column were kept (assist, keyPass, interception, red card, yellow card,opportunity,second yellow card, lost, neutral, won, accurate, Feint, dangerous ball lost, counter attack). The above entries were not helpful and insightful for analysis.

The same reasoning was applied to the *eventName* and *subEventName* columns. Some of the entries were filtered out because they were not insightful and valuable for the model. From *eventName* only the following were kept:(Duel,Foul,Free Kick,Others on the ball,Pass,Shot).Instead in the *subEventName* more rows were used(Ground attacking duel,Ground defending duel,Ground loose ball duel,Hand foul,Late card foul,Out of game foul,Protest,Simulation,Violent Foul,Free kick cross,Penalty,Free kick shot, Throw-in, Acceleration, Clearance, Touch, Cross,Hand pass,Head pass, High pass, Launch, Simple pass, Smart pass, Shot, Air duel, Foul, Free Kick, Corner).

The data-set published by *Wyscout* was already cleaned from dirty records. This is because the great work made by their team. Following the tagging, each match goes through a quality control process: The first phase is automatic (an algorithm is used to prevent most operator errors); The second step of quality control is manual and supervised by quality controllers.

2.4 Feature Engineering

The process of extracting features from unprocessed data and converting them into formats appropriate for machine learning models is known as *feature engineering*. The correct features can make modeling less challenging and enable the pipeline to output results of greater quality, making it an essential step in the machine learning process. After the parsing of the *JSON* files, many columns presented an useless format such as nested dictionaries. This kind of format can not be used to fit a model for this reason all the information were extracted and placed in singular new column. This process was applied to the following columns:

- **Events Position**: *format*: xstart: x, ystart: y, both the values were extracted and added to 2 new columns having the names of the keys.
- **Matches Teams data** *format*: dictionary containing the following keys:
  - "coachId":
  - "formation":
    - "bench": []
    - "lineup": []
    - "substitutions":[]
  - "hasFormation":
  - "score":
  - "scoreET":
  - "SCoreHT"
  - "scoreP":
  - "side":

Every one of the above information were taken and added the newly created columns having as names the keys.

Event Second

As previously stated, one of the column contains the seconds related to the event. Taking into consideration that each entry includes the seconds passed from the first or second half of the game, the column is entirely converted on a minutes base.

Tags

The most challenging part of the *feature engineering* phase regards the Tag column in the *events* table. This information will be essential in modeling since the ML process will use them as variables. The column has the form of a dictionary that contains the tags. For the modeling stage, it is necessary to have a combination of *eventName, subEventName and tags.* A total of 67 variables were selected (2.1)

Features	
Duel.Air.duel.accurate	Others.on.the.ball.Acceleration.not.accurate
Duel.Air.duel.not.accurate	Others.on.the.ball.Clearance.accurate
Duel.Ground.attacking.duel.accurate”	Others.on.the.ball.Clearance.not.accurate
Duel.Ground.attacking.duel.not.accurate”	Others.on.the.ball.Touch.assist
Duel.Ground.defending.duel.accurate	Others.on.the.ball.Touch.counter_attack
Duel.Ground.defending.duel.not.accurate	Others.on.the.ball.Touch.dangerous_ball_lost
Duel.Ground.loose.ball.duel.accurate	Others.on.the.ball.Touch.interception
Duel.Ground.loose.ball.duel.not.accurate	Others.on.the.ball.Touch.opportunity
Foul.Hand.foul.red_card	Pass.Cross.accurate
Foul.Hand.foul.yellow_card	Pass.Cross.assist
Foul.Late.card.foul.yellow_card	Pass.Cross.keyPass
Foul.Foul.red_card	Pass.Cross.not.accurate
Foul.Foul.second_yellow_card	Pass.Hand.pass.accurate
Foul.Foul.yellow_card	Pass.Hand.pass.not.accurate
Foul.Out.of.game.foul.red_card	Pass.Head.pass.accurate
Foul.Out.of.game.foul.second_yellow_card	Pass.Head.pass.assist
Foul.Protest.red_card	Pass.Head.pass.keyPass
Foul.Out.of.game.foul.yellow_card	Pass.Head.pass.not.accurate
Foul.Protest.second_yellow_card	Pass.High.pass.accurate
Foul.Protest.yellow_card	Pass.High.pass.assist
Foul.Simulation.yellow_card	Pass.High.pass.keyPass
Foul.Violent.Foul.red_card	Pass.High.pass.not.accurate
Foul.Violent.Foul.yellow_card	Pass.Launch.accurate
Free.Kick.Free.kick.cross.accurate	Pass.Launch.keyPass
Free.Kick.Free.kick.cross.not.accurate	Pass.Launch.not.accurate
Free.Kick.Free.Kick.accurate	Pass.Simple.pass.accurate
Free.Kick.Free.Kick.not.accurate	Pass.Simple.pass.keyPas
Free.Kick.Penalty.not.accurate	Pass.Simple.pass.not.accurate
Free.Kick.Free.kick.shot.not.accurate	Pass.Smart.pass.accurate
Free.Kick.Free.kick.shot.accurate	Pass.Smart.pass.assist
Free.Kick.Throw.in.accurate	Pass.Smart.pass.keyPass
Free.Kick.Throw.in.not.accurate	Pass.Smart.pass.not.accurate
Others.on.the.ball.Acceleration.accurate	Shot.Shot.accurate
Shot.Shot.not.accurate	

Table 2.1: Features selected for the modelling

Chapter 3

Models

3.1 Player rating

This section will reinterpret the *playerRank model* described in the *"PlayeRank: Data-driven Performance Evaluation and Player Ranking in Soccer via a Machine Learning Approach "* [7], together with its implementation.

To be prepared for a game, a numerical valuation of the overall team rating could be precious. For example, the coach could look at each player's ratings and understand who is the most dangerous. Machine learning can help in the process, but the problem is that the *Wyscout* data set doesn't provide a target variable to build a supervised model. There is no objective and numerical evaluation of the players. *Player Rank* tries to find a solution to this problem by ranking the players based on their performances. For this purpose, all the 67 features described in section 2.4 were used. First, a classification model will try to predict the game's outcome. Secondly, the weights estimated by the model will be extracted and used to rank each player based on his performances.

The following sections will describe every step of the *Player Rank* framework.

3.1.1 Team Performance Extraction

The first phase regards the extraction of the performance of each team in each game. To retrieve this information is enough to group by every match. Every observation of the resulting table will be a match represented by the 67 features.

$$X = [x_1, ..., x_i]$$

$$x_i = \sum_{r=1}^{n} x_i$$

In the above equation Xi is the team performance vector while xi is the i variable.

3.1.2 Feature Weight Extraction

The framework's second step is about classifying the match outcome, which is not the primary objective, but it is fundamental in understanding how much each event type influences the result. The likelihood of winning a game is more strongly influenced by some players' actions throughout a match than by others. For example, a red card will probably negatively affect the outcome. The analysis excludes events such as *Goal* because they can make the model biased. Thus from the model is extracted the vector W of feature weights:

$$W = [w_1, ..., w_i]$$

that represents the importance of each variable on the performance of the team.

3.1.3 Individual Performance Extraction

First, the extraction of each player's (u) performance (r) in each game (m) is needed. It is given by the vector:

$$P_u^m = [x_1, ..., xi]$$

in which the xi represents the i feature. Finally, the vector of weight W will be used to weight each of the xi:

$$r(u, m) = x \sum_{i=1}^{n} w_i \cdot x_i$$

The last step is to get the mean over all the m matches for each player u

$$\overline{r}(u, M) = \frac{1}{R} \sum_{i=1}^{n} r(u, m_i)$$

3.1.4 Application

Using the data set published by *wyscout*, the following section will describe the framework implementation explained in section 3.1. Only the data from the *Italian first division* will be used in this process.

Features Weights Extraction

First, a *Support vector machine* is used as a classification model. It should be trained to predict a binary variable:

- *1: Win*
- *0: Lose/Draw*

The data set used (760X68) was divided into a training set 0.75 and a test set 0.25. As metric the AUC was selected on a 5-fold cross validation. The results are resumed in table 3.1

Metric	AUC	F1	Accuracy	Sensitivity
Result	0.84	0.88	0.85	0.92

Table 3.1: SVM results

Then the weight were extracted from the SVM. Fig 3.1 shows the top 8 positive and negative feature weights. It seams that an accurate shot has very positive impact on the outcome of a game while, for example a not accurate simple pass has a negative impact on it.

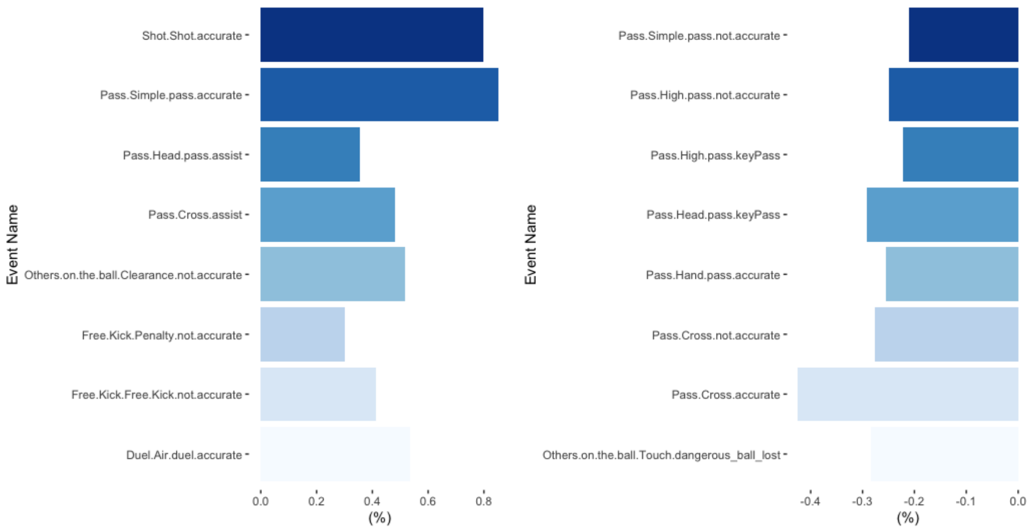

Figure 3.1: Top 8 positive and negative feature weights

Player Ratings

The last step is computing each player's actual rating, as explained in section 3.1.3. The table 3.2 shows some of the best players selected by the model. It looks like the players that played in *Napoli* and *Juventus* are the best ones. In fact, in the *Serie A season 2017/18*, both the teams finished respectively as 2nd and 1st.

Team	Player	Rating
Napoli	Jorginho	0.54
Napoli	Kalidou Koulibaly	0.50
Juventus	Daniele Rugani	0.42
Juventus	Giorgio Chiellini	0.39
Inter	Milan Skriniar	0.38

Table 3.2: 5 of the top player ratings selected by the model

To have a better understanding of the results, fig. 3.2 shows the box plot and distribution of the ratings. The mean is around 0.17, while it looks like only few players are above the 3rd quartile.

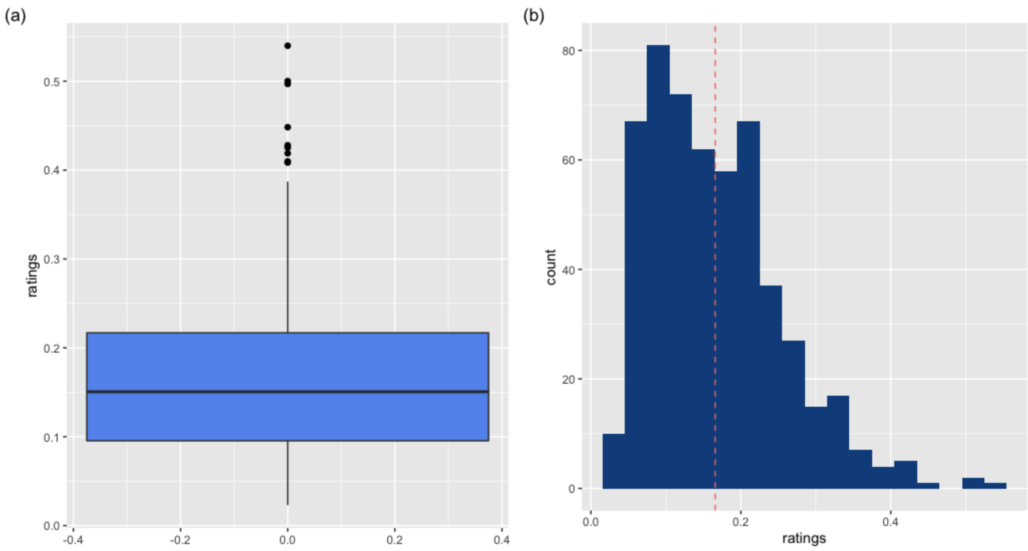

Figure 3.2: Visualization of the results distribution

3.2 Clustering For Role Detection

3.2.1 Introduction To The Purpose Of The Model

The data set published by *wyscout* does not precisely classify each player's role. It only ranks the players according to 3 areas of the field: *defender, midfielder, and forward*. In soccer, there are many more decomposition of those areas, such as central defender and right/left back. Often, the role given at the start of the game does not reflect the player's actual position during the whole game, which is what matters in building a strategy [8]. To make a quick example, going back to Fig.2.7 (b), *Aleksandar Kolarov* is a right back, so he is a defender, and he is supposed to stay more frequently in the defender's area, but looking at the plot it is clear that he could be classified as right-wing. Thus a coach could inform his left wing to help his left back when defending. For this reason, an unsupervised clustering model is implemented to try to find clear areas in the pitch that reflect the roles.

3.2.2 K-mean Clustering explained

Unsupervised machine learning techniques such as clustering are used to find and group comparable data points in larger datasets without regard to the final result. The chosen model is the Kmean clustering which can be resumed in the following steps.
Lloyd's Algorithm

1. *Specify number of clusters K*
2. *Randomly assign each data point to a cluster*
3. *Compute cluster centroids*
4. *Reassign each point to the closest cluster centroid*
5. *Re-compute cluster centroids*
6. *Repeat 4 and 5 until no improvement is made*

3.2.3 Model implementation

As stated in the section 3.2.1, the objective of the clustering model is to find relevant pitch areas that partially reflect the actual roles of the players. For this purpose, the two columns *xstart and ystart* will be used to fit the model. The main objective of the analysis drove this choice. The model does not need to find the characteristics of each field area in terms of event type (even if it can be an interesting analysis). It needs to look for areas that are strongly correlated with each player's position. The second choice regards the data type in terms of single or grouped observations. The results drove the choice. Using the mean position of each player in each game, the model managed to find more defined areas. The last problem is about the only hyperparameter of the model K. It specifies the number of clusters that should be created. The selection of K can be made using different methods. In this case, the choice was driven both by numerical results and the model's objective. It does not make sense to use a small K. Otherwise, only a few non-informative areas will be clustered. As an objective selection, the silhouette score is used, as shown in Fig 3.3.

Silhouette Score

2	3	4	5	6	7	8	9	10
0.4107	0.3884	0.3728	0.3668	0.3667	0.3765	0.3872	0.3842	0.3701

Table 3.3: Silhouette results

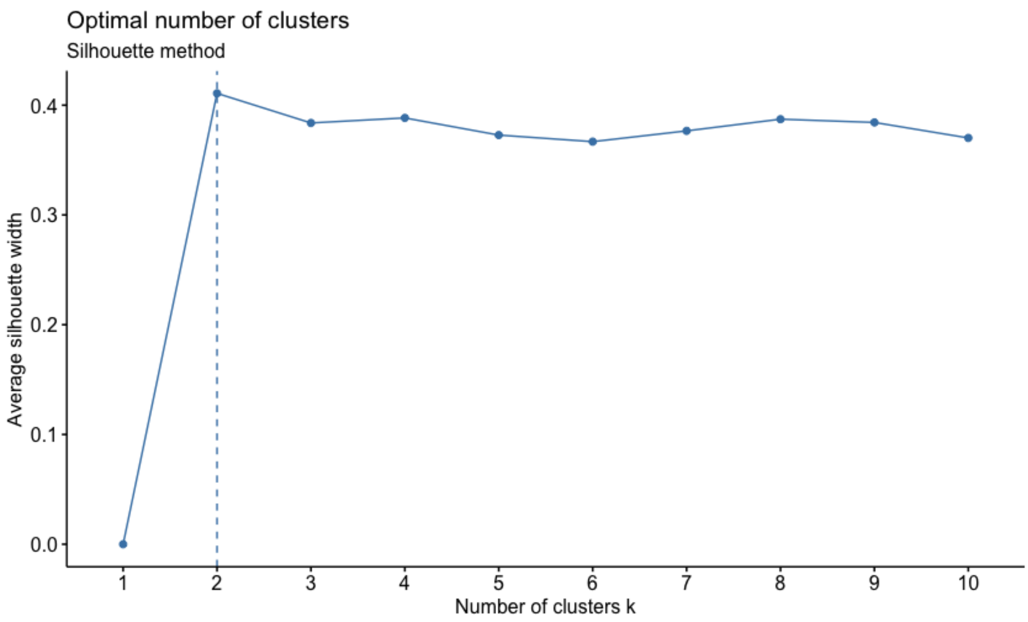

Figure 3.3: Silhouette Score

By looking just at the numbers in table 3.3, it is clear that 2 clusters give the highest score, but it would not make any sense for our analysis to divide

5	6	7	8
Right Back	Left Wing	Striker	Central Midfilder

Table 3.4: Interpretation of the clusters

the field into two sections. Thus by matching the score and the objective, we can choose k=8, which is the second highest score.

Fig 3.4 shows the clustering result using a k=8. The model clearly managed to delimit eight-pitch areas that could reflect the players' roles. The following table gives a possible interpretation of the 8 clusters.

Cluster	1	2	3	4
Interpretation	Left defender	Right Defender	Right Wing	Left Back

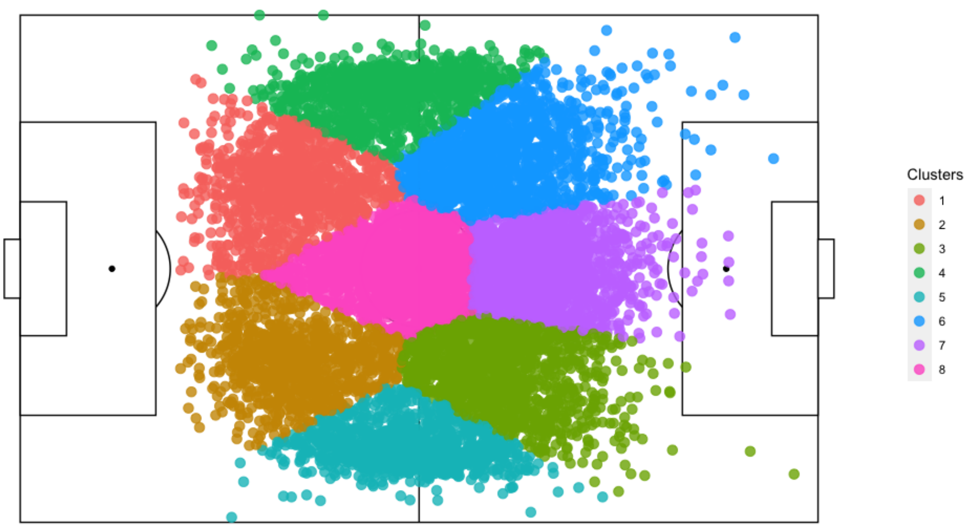

Figure 3.4: K-means clustering result with K=8

3.2.4 Application

There are several possible applications of this clustering model. A coach could need a report about the team he will face next week. Precious information would be to classify each player in one of the 8 clusters, so a more targeted strategy could be implemented. To make an example, it would be insightful to understand where a player plays most of his match so to make, if necessary more density in specific areas. An even more specific application could be to use those 8 clusters to make a network of passes, to understand dangerous areas.

Conclusions

This thesis presented a set of possible solutions for automating several soccer match analysis tasks using a data-driven approach. A static but functional method proved how visualization could help capture insights about every team, from the performance of a single-payer to the analysis of the whole team considered as a complex system. Two machine learning models were also used in the process. The first may help capture information about the rating of each player or team, and the second could help get insightful statistics about every player's actual position during a match. In conclusion, a match analysis powered by data is possible. It is clear that patterns exist in how a player or a team plays in a soccer match. But there are limitations. Since soccer is a sport of creativity and unpredictability, it is challenging to make specific analyses. The hope is that this thesis managed to show the possibilities of using data to automate several tasks in the match analysis, highlighting patterns in this kind of data.

Appendix A

Python and R Code

A.1 Python

A.1.1 JSON parsing

The following part of the python code handle all the parsing of the JSON files together with some pre-processing and feature extraction.

```
import json as js
import pandas as pd
from utilities import *


#TAGs
tags2name='CSV/tags2name.csv'


with open(tags2name) as train_file:
    tags2name_df = pd.read_csv(train_file)


tags2name_df.drop("Description", axis=1, inplace=True)

tags2name_df_pr=tags2name_df.loc[(tags2name_df['Label']=='assist')
    |
     (tags2name_df['Label']=='keyPass') |
         (tags2name_df['Label']=='interception') |
         (tags2name_df['Label']=='opportunity') |
     (tags2name_df['Label']=='red_card') |
         (tags2name_df['Label']=='yellow_card') |
```

```
        (tags2name_df['Label']=='second_yellow_card')|
    (tags2name_df['Label']=='lost') |
        (tags2name_df['Label']=='neutral') |
        (tags2name_df['Label']=='won') |
    (tags2name_df['Label']=='accurate') |
        (tags2name_df['Label']=='not accurate') |
        (tags2name_df['Label']=='Feint') |
    (tags2name_df['Label']=='dangerous_ball_lost')|
    (tags2name_df['Label']=='counter_attack')]


tags2name_dict_2=dict(tags2name_df.values)


tags2name_dict_PR=dict(tags2name_df_pr.values)

#TEAMs
teams = 'teams.json'

with open(teams) as train_file:
    teams_df = pd.read_json(train_file)



teams_dict=dict(teams_df.iloc[:,[2,1]].values)


teams_df.to_csv('/Users/martino/Desktop/dataset
    tesi/CSV/teams.csv')


#PLAYERs
players = 'players.json'

with open(players) as train_file:
    players_df = pd.read_json(train_file)

players_dict=dict_players(players_df)

players_df.to_csv('/Users/martino/Desktop/dataset
    tesi/CSV/players.csv')
```

```
#Next thing to do is to parse the json file of the Events.
#that will be the core of the analysis

#the following functions from the utilities will be applied :
    #cordinates: to extract the x y cortidates from the positions
        column
    #get_names_players: to extract the full name of every player
    #get_names_teams: to extract the name of every name
    #get_tags: to expand the tags column since every events could
        have more than one tag
    #one hot encode: for every tag so to have a matrix (0,1)
        tag,event

#Insted for the DF used to dit the models a combination of event
    name and tag is used to for the matrix
    #filter out some of the useless avents
    #get_tags_for_player_rank: form the matrix


#-------------Italy---------------------
events_Italy='events/events_Italy.json'


with open(events_Italy) as train_file:
    events_Italy_df = pd.read_json(train_file)


cordinates(events_Italy_df)

get_names_players(events_Italy_df,players_dict)

get_names_teams(events_Italy_df,teams_dict)

events_Italy_df_DV=get_tags(events_Italy_df,tags2name_dict_2)

events_Italy_df_DV = pd.DataFrame(events_Italy_df_DV,columns
    =['id','tags'])
events_Italy_df_DV= pd.crosstab(events_Italy_df_DV['id'],
    events_Italy_df_DV['tags'])
events_Italy_df_DV=events_Italy_df_DV.reset_index()
events_Italy_df_DV = pd.merge(events_Italy_df, events_Italy_df_DV,
    how='inner')
events_Italy_df_DV.to_csv('/Users/martino/Desktop/dataset
```

```
    tesi/CSV/CSV_events/events_Italy_DV.csv')

events_Italy_df=
    events_Italy_df.loc[(events_Italy_df['eventName']=='Duel') |
             (events_Italy_df['eventName']=='Foul') |
            (events_Italy_df['eventName']=='Free Kick') |
            (events_Italy_df['eventName']=='Others on the ball')|
            (events_Italy_df['eventName']=='Pass')|
            (events_Italy_df['eventName']=='Shot')]

events_Italy_df=
    events_Italy_df.loc[(events_Italy_df['subEventName']=='Ground
    attacking duel') |
    (events_Italy_df['subEventName']=='Ground defending duel')|
    (events_Italy_df['subEventName']=='Ground loose ball duel')|
    (events_Italy_df['subEventName']=='Hand foul')|
    (events_Italy_df['subEventName']=='Late card foul')|
    (events_Italy_df['subEventName']=='Out of game foul')|
    (events_Italy_df['subEventName']=='Protest')|
    (events_Italy_df['subEventName']=='Simulation')|
    (events_Italy_df['subEventName']=='Violent Foul')|
    (events_Italy_df['subEventName']=='Free kick cross')|
    (events_Italy_df['subEventName']=='Penalty')|
    (events_Italy_df['subEventName']=='Free kick shot')|
    (events_Italy_df['subEventName']=='Throw in')|
    (events_Italy_df['subEventName']=='Acceleration')|
    (events_Italy_df['subEventName']=='Clearance')|
    (events_Italy_df['subEventName']=='Touch')|
    (events_Italy_df['subEventName']=='Cross')|
    (events_Italy_df['subEventName']=='Hand pass')|
    (events_Italy_df['subEventName']=='Head pass')|
    (events_Italy_df['subEventName']=='High pass')|
    (events_Italy_df['subEventName']=='Launch')|
    (events_Italy_df['subEventName']=='Simple pass')|
    (events_Italy_df['subEventName']=='Smart pass')|
    (events_Italy_df['subEventName']=='Shot')|
    (events_Italy_df['subEventName']=='Air duel')|
    (events_Italy_df['subEventName']=='Foul')|
    (events_Italy_df['subEventName']=='Free Kick')|
    (events_Italy_df['subEventName']=='Corner')]

events_Italy_rank_player_df=get_tags_for_player_rank(events_Italy_df
,tags2name_dict_PR)
```

```
events_Italy_rank_player_df =
    pd.DataFrame(events_Italy_rank_player_df,columns =['id','tags'])

events_Italy_rank_player_df=
    pd.crosstab(events_Italy_rank_player_df['id'],
    events_Italy_rank_player_df['tags'])

events_Italy_rank_player_df=events_Italy_rank_player_df.reset_index()

events_Italy_rank_player_df = pd.merge(events_Italy_df,
    events_Italy_rank_player_df, how='inner')

events_Italy_rank_player_df.to_csv('/Users/martino/Desktop/dataset
    tesi/CSV/CSV_events/events_Italy_PR.csv')


matches_Italy='matches/matches_Italy.json'


with open(matches_Italy) as train_file:
    matches_Italy_df = pd.read_json(train_file)

get_info_team(matches_Italy_df)

matches_Italy_df.to_csv('/Users/martino/Desktop/dataset
    tesi/CSV/CSV_matches/matches_Italy.csv')
```

A.2 R Code

A.2.1 Data cleaning and Visualization

The following code regards every aspects of the cleaning and visualization of the data

```
require(tidyverse)
require(dplyr)
library(ggplot2)
library(ggsoccer)
library(raster)
require(ggpubr)
require(forcats)
```

```r
library(soccermatics)

#import the data
events_italy=read.csv(file='CSV/CSV_events/events_Italy.csv',
                      header=T,
                      sep=',',
                      stringsAsFactors=T,
                      dec='.')

#take a look at the data
str(events_italy)
summary(events_italy)

#----------Data cleaning-----------
#eventSec column
#delete useless columns (positions,X,tags,NA.)
events_italy=events_italy%>%dplyr::select(-c(positions,X,tags,NA.))


#convert sec into minutes taking into account the division of
    first and second half

secondH <- events_italy%>% dplyr:: filter (matchPeriod=='2H')
secondH$eventMin <- ((secondH$eventSec)/60)+45

#discretization of 'eventSec' variable for visualization

secondH$timeframe <- cut(secondH$eventMin,
                         breaks =
                             c(45,50,55,60,65,70,75,80,85,90,120),
                         labels=c('45-50','50-55','55-60', '60-65',
                             '65-70' , '70-75','75-80','80-85',
                             '85-90','>90'))
firstH <- events_italy%>% dplyr:: filter (matchPeriod=='1H')
firstH$eventMin <- ((firstH$eventSec)/60)

firstH$timeframe <- cut(firstH$eventMin,
                         breaks = c(0,5, 10, 15, 20, 25
                             ,30,35,40,46,60),
                         labels=c('0-5', '5-10', '10-15', '15-20',
                             '20-25' ,'25-30',
                             '30-35','35-40','40-45',
                                  '>45'))
```

```r

events_italy <- rbind(firstH, secondH)

#make 2 copies for later use
events_italy_PR <- events_italy

events_italy_RD <- events_italy

#----------Data visualization---------
events_italy_DV=read.csv(file='CSV/CSV_events/events_Italy_DV.csv',
                     header=T,
                     sep=',',
                     stringsAsFactors=T,
                     dec='.')
#eventSec column
#delete useless columns (positions,X,tags,NA.)
events_italy_DV=events_italy_DV%>%dplyr::select(-c(positions,X,tags,NA.))

#convert sec into minutes taking into account the division of first and second half
secondH <- events_italy_DV%>% dplyr:: filter (matchPeriod=='2H')
secondH$eventMin <- ((secondH$eventSec)/60)+45

#discretization of 'eventSec' variable for visualization

secondH$timeframe <- cut(secondH$eventMin,
                       breaks = c(45,50,55,60,65,70,75,80,85,90,120),
                       labels=c('45-50','50-55','55-60', '60-65', '65-70' , '70-75','75-80','80-85', '85-90','>90')
)

firstH <- events_italy_DV%>% dplyr:: filter (matchPeriod=='1H')
firstH$eventMin <- ((firstH$eventSec)/60)

firstH$timeframe <- cut(firstH$eventMin,
                      breaks = c(0,5, 10, 15, 20, 25 ,30,35,40,46,60),
                      labels=c('0-5', '5-10', '10-15', '15-20', '20-25' ,'25-30', '30-35','35-40','40-45',
                          '>45')
)
```

```r

events_italy_DV <- rbind(firstH, secondH)
#bar chart for frequency along the game of yellow and red cards
    together with goals

library(RColorBrewer)

mycolors <- colorRampPalette(brewer.pal(8, "Blues"))(20)

red_card_plot <- ggplot(events_italy_DV, aes(x=timeframe ,
    y=red_card, fill=matchPeriod ))+
  geom_bar(stat = 'identity')+
  labs(
      x = "Match Interval", y = "",
      tag = "Fig.1",
      color = "Match Period")+
  scale_fill_manual("Legend", values = c("1H" = "dodgerblue4", "2H"
      = "deepskyblue"))+
   theme(plot.title = element_text(size=18))

yellow_card_plot <- ggplot(events_italy_DV, aes(x=timeframe,
    y=yellow_card, fill=matchPeriod))+
  geom_bar(stat = 'identity')+
  labs(
      x = "Match Interval", y = "",
      tag = "Fig.2",
      color = "Match Period")+
  scale_fill_manual("Legend", values = c("1H" = "dodgerblue4", "2H"
      = "deepskyblue"))+
  theme(plot.title = element_text(size=18))


Goal_plot <- ggplot(events_italy_DV, aes(x=timeframe, y=Goal,
    fill=matchPeriod))+
  geom_bar(stat = 'identity')+
  labs(
      x = "Match Interval", y = "",
      tag = "Fig.3",
      color = "Match Period")+
  scale_fill_manual("Legend", values = c("1H" = "dodgerblue4", "2H"
      = "deepskyblue"))+
  theme(plot.title = element_text(size=18))

```

```r
figure2 <- ggarrange(red_card_plot,yellow_card_plot,Goal_plot, ncol = 1, nrow = 3)


#Bar chart for event frequency

events_italy_DV %>%
  count(eventName) %>%
  mutate(perc = n / nrow(events_italy_DV)) -> tips2

tips2$eventName <- factor(tips2$eventName,
                          levels = c('Goalkeeper leaving line','Offside','Save attempt','Shot',
                                     'Foul','Interruption','Free Kick','Others on the ball','Duel','Pass'))

levels(tips2$eventName)
levels(events_italy_DV$eventName)

ggplot(tips2, aes(x = eventName, y = perc, fill=eventName)) +
  geom_bar(stat = "identity")+coord_flip()+
  scale_fill_manual(values = mycolors)+
  theme(panel.background = element_rect(fill = "white"),legend.position="none",plot.title = element_text(size=18))+
  labs(x = "Event Name", y = "(%)")

# Pitch heatmap for type of event

pass_heatmap<- ggplot() +
                  annotate_pitch( fill= 'white', colour = "black") +
                  theme_pitch() +
theme(panel.background = element_rect(fill = "White"),legend.position="none")+
geom_density_2d_filled(data = events_italy_DV%>%dplyr::filter(eventName=='Pass'),
aes(x = x_start, y = y_start ),alpha=0.8 )+scale_fill_manual(values = mycolors)+
labs(title = "Pass",
x = "", y = "",
tag = "(a)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))
```

```
shot_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(eventName=='Shot'),
                         aes(x = x_start, y = y_start ),alpha=0.6 )+
  scale_fill_brewer(palette="Blues")+
  labs(title = "Shots",
       x = "", y = "",
       tag = "(b)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))


duel_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(eventName=='Duel'),
                         aes(x = x_start, y = y_start ),alpha=0.8 )+
  scale_fill_manual(values = mycolors)+
  labs(title = "Duel",
       x = "", y = "",
       tag = "(c)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))

foul_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(eventName=='Foul'),
                         aes(x = x_start, y = y_start ),alpha=0.8 )+
  scale_fill_manual(values = mycolors)+
  labs(title = "Fauls",
       x = "", y = "",
       tag = "(d)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))
```

```
figure <- ggarrange(pass_heatmap, shot_heatmap, duel_heatmap,
    foul_heatmap,
                     ncol = 2, nrow = 2)

#focus on a single game

focus_DV <- events_italy_DV%>%dplyr::filter(matchId==2576300)

focus_goal <- focus_DV%>%dplyr::filter(Goal==1 & eventName=='Shot')

#Goals
focus_goal_plot<- ggplot() +
  annotate_pitch( fill= 'dodgerblue4', colour = "white") +
  theme_pitch() +
  geom_point(data = focus_goal,aes(x = x_start, y = y_start, fill =
      t_name),
            shape = 21,
            size = 3)+
  labs(
       x = "", y = "")+
  theme(plot.title = element_text(size = 14,hjust = 0.5),
        panel.background = element_rect(fill = "White"))+
  scale_fill_manual("Team", values = c( "Roma" =
      "orange","Chievo"="yellow"))

focus_shot_r <- focus_DV%>%dplyr::filter(eventName=='Shot' &
    t_name=='Roma')

#shots Roma
focus_shot_plot_r<- ggplot() +
  annotate_pitch( fill= 'dodgerblue4', colour = "white") +
  theme_pitch() +
  geom_point(data = focus_shot_r,aes(x = x_start, y = y_start, fill
      = t_name),shape = 21,size = 3)+
  labs( x = "", y = "",
       tag = "(a)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5),
        panel.background = element_rect(fill = "White"))+
  scale_fill_manual("Team", values = c( "Roma" = "orange"))

focus_shot_c <- focus_DV%>%dplyr::filter(eventName=='Shot' &
```

```
    t_name=='Chievo')

#shots Chievo
focus_shot_plot_c<- ggplot() +
  annotate_pitch( fill= 'dodgerblue4', colour = "white") +
  theme_pitch() +
  geom_point(data = focus_shot_c,aes(x = x_start, y = y_start, fill
      = t_name),
             shape = 21,
             size = 3)+
  labs(
       x = "", y = "",
       tag = "(b)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5),
        panel.background = element_rect(fill = "White"))+
  scale_fill_manual("Team", values = c( "Chievo" = "yellow"))

figure3 <- ggarrange(focus_shot_plot_r, focus_shot_plot_c,
                     ncol = 2, nrow = 1)

focus_chievo <- events_italy_DV%>%dplyr::filter(teamId==3165 &
    Goal==1 & eventName!='Save attempt')


#chievo heat map shots
mycolors2 <- colorRampPalette(brewer.pal(8, "Blues"))(12)

focus_goal_chievo_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  scale_fill_manual(values = mycolors2)+
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(teamId==3165 & Goal==1&
                          eventName!='Save attempt'),
                       aes(x = x_start, y = y_start ),alpha=0.8 )+
  labs( x = "", y = "",
       tag = "(a)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))

mycolors3 <- colorRampPalette(brewer.pal(8, "Greens"))(11)

```

```
#kolarov heatmap
focus_kolarov_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  scale_fill_manual(values = mycolors3)+
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(p_name=='Aleksandar Kolarov'),
                              aes(x = x_start, y = y_start ),alpha=0.8 )+
  labs(
    x = "", y = "",
    tag = "(b)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))

#pass chievo heatmap
focus_pass_chievo_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  scale_fill_manual(values = mycolors2)+
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(teamId==3165 &
      eventName!='pass'), aes(x = x_start, y = y_start ),alpha=0.8
      )+
  labs( x = "", y = "",
    tag = "(c)")+
  theme(plot.title = element_text(size = 14,hjust = 0.5))

focus_pass_Roma_heatmap<- ggplot() +
  annotate_pitch( fill= 'white', colour = "black") +
  theme_pitch() +
  scale_fill_manual(values = mycolors3)+
  theme(panel.background = element_rect(fill =
      "White"),legend.position="none")+
  geom_density_2d_filled(data =
      events_italy_DV%>%dplyr::filter(teamId==3158 &
      eventName!='pass'),
                              aes(x = x_start, y = y_start ),alpha=0.8 )+
  labs(
    x = "", y = "",
    tag = "(d)")+
```

```
  theme(plot.title = element_text(size = 14,hjust = 0.5))


figure5 <- ggarrange(focus_goal_chievo_heatmap,
    focus_kolarov_heatmap, focus_pass_chievo_heatmap,
    focus_pass_Roma_heatmap,
                    ncol = 2, nrow = 2)

#Network analysis

match_italy_NA=read.csv(file='CSV/CSV_matches/matches_Italy.csv',
                        header=T,
                        sep=',',
                        stringsAsFactors=T,
                        dec='.')

events_italy_NA <-
    focus_DV%>%dplyr::select(c(eventName,playerId,matchId,
teamId,eventMin,x_start,y_start,x_end,y_end))


events_italy_NA_try <-
    events_italy_NA%>%dplyr::filter(eventName=='Pass')


#build the table

i=1
r=2
possesso <- data.frame(NA_col = rep(NA, nrow(events_italy_NA_try)))

for(elm in events_italy_NA_try$teamId){

  if(r<nrow(events_italy_NA_try)+1){
  if(elm==events_italy_NA_try$teamId[r]){

    possesso[ i,] <- NA

  }
  else{
    possesso[ i, ] <- 'interrupt'
  }
  i <- i+1
```

```
  r <- r+1
  }
}

events_italy_NA_try$possesso <- possesso$NA_col

#find the receiver
i=1
r=2
ricevitori <- data.frame(NA_col = rep(NA, nrow(events_italy_NA_try)))

for (elm in events_italy_NA_try$possesso ) {
  if( is.na(elm) ){

    ricevitori[ i,] <- events_italy_NA_try$playerId[r]
  }
  else{

    ricevitori[ i,] <- NA
  }
  i <- i+1
  r <- r+1
}

events_italy_NA_try$receiver <- ricevitori$NA_col

#find the weights

events_italy_NA_2 <- events_italy_NA_try%>%
dplyr::select(c(playerId,receiver,teamId))%>%
drop_na()%>%
  dplyr::filter(teamId==3158)

events_italy_NA_2 <-data.frame( events_italy_NA_2%>%dplyr::group_by(playerId,receiver)%>%
dplyr::summarise(weights=n())%>%
  rename(from=playerId,to=receiver))

#let's try with a match==2575959
edges <-events_italy_NA_2%>%dplyr::filter(!(from=='340019'|from=='25405'|
from=='92966'|to=='340019'|to=='25405'|to=='92966'))
```

```
edges <- edges %>%
  filter(!(from == to))

edges$weights <- (edges$weights - min(edges$weights)) /
    (max(edges$weights) - min(edges$weights))

str(edges)

nodes <- data.frame(events_italy_NA_2%>%
dplyr::group_by(from)%>%
dplyr::summarise(x=n()))%>%
dplyr::select(-c(x))%>%dplyr::rename(id=from)%>%
  filter(!(id=='340019'|id=='25405'|id=='92966'))

players_NA=read.csv(file='CSV/players.csv',
                        header=T,
                        sep=',',
                        stringsAsFactors=T,
                        dec='.')

players_NA <- players_NA%>%dplyr::select(wyId,
    shortName)%>%rename(id=wyId)


nodes_names<-merge(x=nodes,y=players_NA,by="id",all.x=TRUE)

#build the network

library(network)

routes_network <- network(edges, vertex.attr = nodes,
matrix.type = "edgelist", ignore.eval = FALSE, loops = TRUE)

library(GGally)
library(network)
library(sna)
library(ggplot2)

position_NA <-
    focus_DV%>%dplyr::select(c(playerId,x_start,y_start))%>%
dplyr::group_by(playerId)%>%
  dplyr::summarise(x_mean=mean(x_start),y_mean=mean(y_start))%>%
```

```
  dplyr::rename(id=playerId)

nodes_names_pos<-merge(x=nodes_names,y=position_NA,by="id",all.x=TRUE)

formation_NA = as.matrix(data.frame(x = nodes_names_pos$x_mean , y
    = nodes_names_pos$y_mean ))

roma_network<- ggnet2(routes_network,mode=formation_NA ,size =
    12,edge.size = "weights", label =nodes_names$shortName,color =
    "cornsilk3",
       label.alpha = 0.8,label.color =
           "black",edge.color='cornsilk4')+
  theme(panel.background = element_rect(fill = "white"))

#Distributions variables
goals <- events_italy_DV %>% dplyr::filter(!(eventName=='Save
    attempt'))%>%
  dplyr::filter(Goal==1)%>%dplyr::group_by(matchId)%>%dplyr::summarise(n=n())

hist_goals <- ggplot(goals,aes(x=n))+
  geom_histogram(binwidth=1,fill='dodgerblue4')+
  geom_vline(data=goals, aes(xintercept=mean(goals$n), color="red"),
             linetype="dashed")+
  theme(legend.position="none")+
  labs(tag = "(a)")


box_plot_goals <-
    ggplot(goals,aes(y=n))+geom_boxplot(outlier.colour="black",
    outlier.shape=16,outlier.size=2,notch=FALSE,
    fill='cornflowerblue')+
  labs(tag = "(b)")

pass_hist <- events_italy_DV%>%dplyr::filter(eventName=='Pass'&
    accurate==1)%>%
  dplyr::group_by(matchId)%>%dplyr::summarise(n=n())

hist_pass <- ggplot(pass_hist,aes(x=n))+
  geom_histogram(aes(y=..density..), colour="dodgerblue4",
      fill="dodgerblue4")+
  geom_density(alpha=.1, fill="orange") +
  geom_vline(data=goals, aes(xintercept=mean(pass_hist$n),
      color="red"),
```

```
            linetype="dashed")+
  theme(legend.position="none")+
  labs(tag = "(c)")

pass_hist_not_acc <- events_italy_DV%>%dplyr::filter(eventName=='Pass'& not.accurate==1)%>%
  dplyr::group_by(matchId)%>%dplyr::summarise(n=n())

hist_pass_not_acc <- ggplot(pass_hist_not_acc,aes(x=n))+
  geom_histogram(aes(y=..density..), colour="cornflowerblue", fill="cornflowerblue")+
  geom_density(alpha=.1, fill="red") +
  geom_vline(data=goals, aes(xintercept=mean(pass_hist_not_acc$n), color="orange"),
            linetype="dashed")+
  theme(legend.position="none")+
  labs(tag = "(d)")

figure <- ggarrange(hist_goals, box_plot_goals, hist_pass, hist_pass_not_acc,
                    ncol = 2, nrow = 2)

mean(pass_hist_not_acc$n)
```

A.2.2 Supervised Model(SVM)

The following section shows the code of modelling phase regarding the classification task.

```


#take a look ate the variables and choose which one to exclude
str(events_italy_PR)
summary(events_italy_PR)

#exclude the useless variable for this analysis (we only need the IDs of playes,
#teams and match together with the events
events_italy_PR=events_italy_PR%>%dplyr::select(-c(eventId,subEventName, eventName, matchPeriod, eventSec,subEventId,id,x_start,y_start,x_end,y_end,timeframe))
```

```
#group by matchId and playerID in order to get the sum of each event for every player in every match
events_italy_PR_players=events_italy_PR%>%dplyr::select(-c(p_name,t_name))

events_italy_PR_players <- events_italy_PR_players%>% group_by(matchId,playerId,teamId)%>%summarise_all(list(sum))

#by analysing the staitsics of each column we can exclude the some of them which are useless
#summary(events_italy_PR_players)
#str(events_italy_PR_players)

events_italy_PR_players=events_italy_PR_players%>%
  dplyr::select(c(matchId,playerId,teamId,
  Duel.Air.duel.accurate,Duel.Air.duel.not.accurate,
  Duel.Ground.attacking.duel.accurate,
   Duel.Ground.attacking.duel.not.accurate,
   Duel.Ground.defending.duel.accurate,
   Duel.Ground.defending.duel.not.accurate,
   Duel.Ground.loose.ball.duel.accurate,
   Duel.Ground.loose.ball.duel.not.accurate,
   Foul.Hand.foul.red_card,Foul.Hand.foul.yellow_card,
   Foul.Late.card.foul.yellow_card,Foul.Foul.red_card, Foul.Foul.second_yellow_card,
   Foul.Foul.yellow_card,Foul.Out.of.game.foul.red_card , Foul.Out.of.game.foul.second_yellow_card,
   Foul.Out.of.game.foul.yellow_card,Foul.Protest.red_card ,Foul.Protest.second_yellow_card,
   Foul.Protest.yellow_card, Foul.Simulation.yellow_card,Foul.Violent.Foul.red_card,
   Foul.Violent.Foul.yellow_card,Free.Kick.Free.kick.cross.accurate, Free.Kick.Free.kick.cross.not.accurate,
   Free.Kick.Free.Kick.accurate,Free.Kick.Free.Kick.not.accurate,
   Free.Kick.Penalty.not.accurate,
   Free.Kick.Free.kick.shot.not.accurate,Free.Kick.Free.kick.shot.accurate,Free.Kick.T
   Free.Kick.Throw.in.not.accurate,
   Others.on.the.ball.Acceleration.accurate,
   Others.on.the.ball.Acceleration.not.accurate, Others.on.the.ball.Clearance.accurate,
   Others.on.the.ball.Clearance.not.accurate,
   Others.on.the.ball.Touch.assist,
```

```
  Others.on.the.ball.Touch.counter_attack,
  Others.on.the.ball.Touch.dangerous_ball_lost,
  Others.on.the.ball.Touch.interception,
  Others.on.the.ball.Touch.opportunity,Pass.Cross.accurate,
  Pass.Cross.assist,Pass.Cross.keyPass,
     Pass.Cross.not.accurate,Pass.Hand.pass.accurate,
  Pass.Hand.pass.not.accurate,
  Pass.Head.pass.accurate,Pass.Head.pass.assist,
  Pass.Head.pass.keyPass ,Pass.Head.pass.not.accurate,
  Pass.High.pass.accurate,Pass.High.pass.assist,
  Pass.High.pass.keyPass,Pass.High.pass.not.accurate,
  Pass.Launch.accurate,Pass.Launch.keyPass,Pass.Launch.not.accurate,
  Pass.Simple.pass.accurate,
  Pass.Simple.pass.keyPass,
  Pass.Simple.pass.not.accurate,Pass.Smart.pass.accurate,
     Pass.Smart.pass.assist,
  Pass.Smart.pass.keyPass, Pass.Smart.pass.not.accurate,
  Shot.Shot.accurate,Shot.Shot.not.accurate))

col_names <- colnames(events_italy_PR_players )
#same thing for the team dataset
#group by matchId and teamID in order to get the sum of each event
   for every team in every match
events_italy_PR_teams=events_italy_PR%>%dplyr::select(-c(playerId,p_name,t_name))

events_italy_PR_teams <-
   events_italy_PR_teams%>%group_by(matchId,teamId)%>%summarise_all(list(sum))

#get the binary variable for winner
match_italy_PR=read.csv(file='CSV/CSV_matches/matches_Italy.csv',
                        header=T,
                        sep=',',
                        stringsAsFactors=T,
                        dec='.')


match_italy_PR <- match_italy_PR%>%dplyr:: select(c(wyId ,
   winner))%>%dplyr:: rename(matchId=wyId)


events_italy_PR_teams <-
   merge(events_italy_PR_teams,match_italy_PR, by='matchId'
   ,all.x=TRUE)
```

```
events_italy_PR_teams$outcome <- ifelse(events_italy_PR_teams$winner==events_italy_PR_teams$teamId,1,0)

#by analysing the staitsics of each column we can exclude the some of them which are useless
events_italy_PR_teams=events_italy_PR_teams%>%
  dplyr::select(c(matchId,teamId,outcome,Duel.Air.duel.accurate,
  Duel.Air.duel.not.accurate,Duel.Ground.attacking.duel.accurate,
   Duel.Ground.attacking.duel.not.accurate,
   Duel.Ground.defending.duel.accurate,
   Duel.Ground.defending.duel.not.accurate,
   Duel.Ground.loose.ball.duel.accurate,
   Duel.Ground.loose.ball.duel.not.accurate,
   Foul.Hand.foul.red_card,
   Foul.Hand.foul.yellow_card,
   Foul.Late.card.foul.yellow_card,Foul.Foul.red_card, Foul.Foul.second_yellow_card,
   Foul.Foul.yellow_card,Foul.Out.of.game.foul.red_card , Foul.Out.of.game.foul.second_yellow_card,
   Foul.Out.of.game.foul.yellow_card,Foul.Protest.red_card ,Foul.Protest.second_yellow_card,
   Foul.Protest.yellow_card, Foul.Simulation.yellow_card,Foul.Violent.Foul.red_card,
   Foul.Violent.Foul.yellow_card,Free.Kick.Free.kick.cross.accurate, Free.Kick.Free.kick.cross.not.accurate,
   Free.Kick.Free.Kick.accurate,Free.Kick.Free.Kick.not.accurate,
   Free.Kick.Penalty.not.accurate,Free.Kick.Free.kick.shot.not.accurate,
   Free.Kick.Free.kick.shot.accurate,
   Free.Kick.Throw.in.accurate,Free.Kick.Throw.in.not.accurate,
   Others.on.the.ball.Acceleration.accurate,
   Others.on.the.ball.Acceleration.not.accurate, Others.on.the.ball.Clearance.accurate,
   Others.on.the.ball.Clearance.not.accurate,
   Others.on.the.ball.Touch.assist,Others.on.the.ball.Touch.counter_attack,
   Others.on.the.ball.Touch.dangerous_ball_lost,
   Others.on.the.ball.Touch.interception,
   Others.on.the.ball.Touch.opportunity,Pass.Cross.accurate,
   Pass.Cross.assist,Pass.Cross.keyPass, Pass.Cross.not.accurate,Pass.Hand.pass.accurate,
   Pass.Hand.pass.not.accurate,
   Pass.Head.pass.accurate,Pass.Head.pass.assist,
   Pass.Head.pass.keyPass ,Pass.Head.pass.not.accurate,
```

```
    Pass.High.pass.accurate,Pass.High.pass.assist,
    Pass.High.pass.keyPass,Pass.High.pass.not.accurate,
    Pass.Launch.accurate,Pass.Launch.keyPass,
    Pass.Launch.not.accurate,Pass.Simple.pass.accurate,
    Pass.Simple.pass.keyPass,
    Pass.Simple.pass.not.accurate,Pass.Smart.pass.accurate,
        Pass.Smart.pass.assist,
    Pass.Smart.pass.keyPass, Pass.Smart.pass.not.accurate,
    Shot.Shot.accurate,Shot.Shot.not.accurate))

#delete players not present in the player_df

players_PR=read.csv(file='CSV/players.csv',
                      header=T,
                      sep=',',
                      stringsAsFactors=T,
                      dec='.')

players_PR_prov <-
    players_PR%>%dplyr::select(c(firstName,lastName,wyId))%>%rename(playerId=wyId)

players_PR <- players_PR%>%dplyr::select(c(wyId))%>%dplyr::
    rename(playerId=wyId)

events_italy_PR_players <-
    merge(events_italy_PR_players,players_PR, by='playerId' )

#------Feature weights extraction----
#standardize the data

df_standard <-
    events_italy_PR_teams%>%dplyr::select(-c(teamId,matchId,outcome))

df_standard <- data.frame(scale(df_standard))

df_standard$outcome <- as.factor(events_italy_PR_teams$outcome)

#------------- Fit the SVM linear classifier------------

#train and test set

library(caTools)

```

```
set.seed(123)
split = sample.split(df_standard$outcome, SplitRatio = 0.75)

df_standard <- df_standard %>% mutate(outcome =
    factor(outcome,labels = make.names(levels(outcome))))

training_set = subset(df_standard, split == TRUE)
test_set = subset(df_standard, split == FALSE)

library(caret)

# Define fitControl
fitControl <- trainControl(
  method = "cv",
  number = 5,
  classProbs = TRUE,
  summaryFunction = twoClassSummary
)

# set random seed and run the model

set.seed(321)
svmFit1 <- train(x = training_set[-68] ,
                 y=training_set$outcome,
                 method = "svmLinear",
                 trControl = fitControl,
                 preProc = c("center","scale"),
                 metric="ROC" )

y_pred_2 = predict(svmFit1, newdata = test_set[-68])

confusionMatrix(y_pred_2,test_set[, 68])
#Sensitivity : 0.9224
#Accuracy : 0.8474
#roc:0.84

library("MLmetrics")

f1_svm<- F1_Score(y_pred_2, test_set[, 68], positive =
    NULL)#0.8806584

#get weights

```

```r
coefs <- svmFit1$finalModel@coef[[1]]

mat <- svmFit1$finalModel@xmatrix[[1]]

weights2 <- t(coefs %*% mat)

col_names_weights <- colnames(training_set)[-68]

#visualize weights

#positive weights
weights_dv <- data.frame(col_names_weights,weights2)

weights_dv <-data.frame(weights_dv[order(-weights2),])[1:8,]

weights_pos<- ggplot(weights_dv, aes(x = col_names_weights, y =
    weights2, fill=col_names_weights))+
  scale_fill_brewer(palette="Blues") +
  geom_bar(stat = "identity")+coord_flip()+
  theme(panel.background = element_rect(fill =
      "white"),legend.position="none",plot.title =
      element_text(size=18))+
  labs(x = "Event Name", y = "(%)")

#megative weight
weights_dv_2 <- data.frame(col_names_weights,weights2)

weights_dv_2 <-data.frame(weights_dv_2[order(weights2),])[1:8,]

weights_neg <- ggplot(weights_dv_2, aes(x = col_names_weights, y =
    weights2, fill=col_names_weights))+
  scale_fill_brewer(palette="Blues") +
  geom_bar(stat = "identity")+coord_flip()+
  theme(panel.background = element_rect(fill =
      "white"),legend.position="none",plot.title =
      element_text(size=18))+
  labs(x = "Event Name", y = "(%)")

figure6 <- ggarrange(weights_pos,weights_neg, ncol = 2, nrow = 1)


#---- Player Rating Phase -------

```

```
#dot product vector of players performances * vector of feature
    weights
events_italy_PR_players_rating <-
    events_italy_PR_players%>%dplyr::select(-c(playerId,matchId,teamId))


events_italy_PR_players_rating_M <-
    data.matrix(events_italy_PR_players_rating)

weighted_events_italy_PR_players <-
    events_italy_PR_players_rating_M %*%weights2

range01 <- function(x){(x-min(x))/(max(x)-min(x))}

weighted_events_italy_PR_players_st <-
    data.frame(perfomance_for_match=range01(weighted_events_italy_PR_players)
    )

weighted_events_italy_PR_players_st$matchId <-
    events_italy_PR_players$matchId

weighted_events_italy_PR_players_st$playerId <-
    events_italy_PR_players$playerId

library(qcc)
events_italy_PR_prov <-
    events_italy_PR%>%dplyr::select(c(playerId,p_name))%>%distinct()

player_ratings <- weighted_events_italy_PR_players_st%>%
  dplyr:: select(c(playerId,perfomance_for_match))%>%
      dplyr::group_by(playerId)%>%
  dplyr::summarise(ratings=mean(perfomance_for_match))

player_ratings <-
    merge(x=player_ratings,y=events_italy_PR_prov,by='playerId',all.x=TRUE
    )

#visualize the results

ratings_plot <- ggplot(data=player_ratings, aes(y=ratings))+
  geom_boxplot(outlier.colour="black", outlier.shape=16,
             outlier.size=2, notch=FALSE,
             fill='cornflowerblue')+theme(legend.position="none")+
```

```
  labs(tag = "(a)")

hist_goals <- ggplot(player_ratings,aes(x=ratings))+
  geom_histogram(binwidth=0.03,fill='dodgerblue4')+
  geom_vline(data=player_ratings,
      aes(xintercept=mean(player_ratings$ratings), color="red"),
             linetype="dashed")+
  theme(legend.position="none")+
  labs(tag = "(b)")


figure7 <- ggarrange(ratings_plot,hist_goals, ncol = 2, nrow = 1)
```

A.2.3 Unsupervised Model (K-means)

```

#-------Clustering For Role detection----

library(plotly)
library(factoextra)
library(NbClust)
library(class)
library(MASS)
require(pROC)
library(ISLR)
library(RColorBrewer)
library(wesanderson)

#data cleaning
events_italy_RD=events_italy_RD%>%
  dplyr::select(c(matchId,teamId,playerId,x_start,
      y_start,Duel.Air.duel.accurate,Duel.Air.duel.not.accurate,
  Duel.Ground.attacking.duel.accurate,
      Duel.Ground.attacking.duel.not.accurate,
      Duel.Ground.defending.duel.accurate,
      Duel.Ground.defending.duel.not.accurate,
      Duel.Ground.loose.ball.duel.accurate,
      Duel.Ground.loose.ball.duel.not.accurate,
      Foul.Hand.foul.red_card,Foul.Hand.foul.yellow_card,
      Foul.Late.card.foul.yellow_card,Foul.Foul.red_card,
          Foul.Foul.second_yellow_card,
```

```
      Foul.Foul.yellow_card,Foul.Out.of.game.foul.red_card , Foul.Out.of.game.foul.second_yellow_card,
      Foul.Out.of.game.foul.yellow_card,Foul.Protest.red_card ,Foul.Protest.second_yellow_card,
      Foul.Protest.yellow_card, Foul.Simulation.yellow_card,Foul.Violent.Foul.red_card,
      Foul.Violent.Foul.yellow_card,Free.Kick.Free.kick.cross.accurate, Free.Kick.Free.kick.cross.not.accurate,
      Free.Kick.Free.Kick.accurate,Free.Kick.Free.Kick.not.accurate,
      Free.Kick.Penalty.not.accurate,
      Free.Kick.Free.kick.shot.not.accurate,Free.Kick.Free.kick.shot.accurate,
      Free.Kick.Throw.in.accurate,
      Free.Kick.Throw.in.not.accurate,Others.on.the.ball.Acceleration.accurate,
      Others.on.the.ball.Acceleration.not.accurate, Others.on.the.ball.Clearance.accurate,
      Others.on.the.ball.Clearance.not.accurate,
      Others.on.the.ball.Touch.assist,Others.on.the.ball.Touch.counter_attack,
      Others.on.the.ball.Touch.dangerous_ball_lost,
      Others.on.the.ball.Touch.interception,
      Others.on.the.ball.Touch.opportunity,Pass.Cross.accurate,
      Pass.Cross.assist,Pass.Cross.keyPass, Pass.Cross.not.accurate,Pass.Hand.pass.accurate,
      Pass.Hand.pass.not.accurate,
      Pass.Head.pass.accurate,Pass.Head.pass.assist,
      Pass.Head.pass.keyPass ,Pass.Head.pass.not.accurate,
      Pass.High.pass.accurate,Pass.High.pass.assist,
      Pass.High.pass.keyPass,Pass.High.pass.not.accurate,
      Pass.Launch.accurate,Pass.Launch.keyPass,
      Pass.Launch.not.accurate,Pass.Simple.pass.accurate,
      Pass.Simple.pass.keyPass,
      Pass.Simple.pass.not.accurate,Pass.Smart.pass.accurate, Pass.Smart.pass.assist,
      Pass.Smart.pass.keyPass, Pass.Smart.pass.not.accurate,Shot.Shot.accurate,Shot.Shot.not.accurate))


#as_factor
events_italy_RD$playerId <- as.factor(events_italy_RD$playerId)

events_italy_RD$teamId <- as.factor(events_italy_RD$teamId)


#take into account only the usefull type of events
```

```
coordi=events_italy_RD %>%
  dplyr::select(c(x_start, y_start, playerId,
      matchId))%>%dplyr::group_by(matchId,playerId)%>%
  dplyr::summarise(avg_pos_X = mean(x_start), avg_pos_Y=
      mean(y_start), .groups = 'drop')%>%
  dplyr::filter(avg_pos_X>20)%>%dplyr::select(-c(playerId,matchId))


#k=8
# Silhouette method
#Measures the quality of a clustering and determines how well each
    point lies within its cluster.
silhouette <- fviz_nbclust(coordi, kmeans, method = "silhouette") +
              labs(subtitle = "Silhouette method")

silhouette_score <- data.frame(silhouette[["data"]][["y"]])

km_1 = kmeans(coordi, 8, nstart = 1, iter.max = 1e2)


coordi$Clusters <- as.factor( km_1$cluster)

par(mfrow=c(1,1))


p <- ggplot(coordi, aes(x=avg_pos_X, y=avg_pos_Y)) +
  annotate_pitch(fill = "white", colour = "black") +
  theme_pitch()+
  geom_point(aes(col =Clusters),alpha=0.8 ,size =3) +
  scale_fill_manual("Legend")+
  coord_fixed()

results <- data.frame(coordi$avg_pos_X,coordi$avg_pos_Y)
results <- data.frame(NA_col = rep(NA, nrow(coordi)))
```

Bibliography

[1] Rossi A, Pappalardo L, Cintia P, Iaia FM, Fernàndez J, Medina D (2018) Effective injury forecasting in soccer with GPS training data and machine learning. PLoS ONE 13(7): e0201264. https://doi.org/10.1371/journal.pone.0201264

[2] Prakoso, Muhammad Lumintuarso, Ria. (2021). Analysis of the Use of Statistical Data in the Formulation of Strategies, Tactics and Evaluation of Football Matches. 10.2991/ahsr.k.210707.009.

[3] Ballesta, Christian Garcia-Romero, Jerónimo José Carlos, Fernández-García Alvero Cruz, Jose Ramon. (2015). Current methods of soccer match analysis. Revista Internacional de Medicina y Ciencias de la Actividad Fisica y del Deporte. 15. 785-803.

[4] Pappalardo, L., Cintia, P., Rossi, A. et al. A public data set of spatio-temporal match events in soccer competitions. Sci Data 6, 236 (2019). https://doi.org/10.1038/s41597-019-0247-7

[5] Buldú JM, Busquets J, Martínez JH, Herrera-Diestra JL, Echegoyen I, Galeano J and Luque J (2018) Using Network Science to Analyse Football Passing Networks: Dynamics, Space, Time, and the Multilayer Nature of the Game. Front. Psychol. 9:1900. doi: 10.3389/fpsyg.2018.01900

[6] Cintia, P., Rinzivillo, S. Pappalardo, L. Network-based Measures for Predicting the Outcomes of Football Games. Proceedings of the 2nd Workshop on Machine Learning and Data Mining for Sports Analytics (MLSA), 46–54 (2015).

[7] Luca Pappalardo, Paolo Cintia, Paolo Ferragina, Emanuele Massucco, Dino Pedreschi, and Fosca Giannotti. 2019. PlayeRank: Data-driven Performance Evaluation and Player Ranking in Soccer via a Machine Learning Approach. ACM Trans. Intell. Syst. Technol. 10, 5, Article 59 (September 2019), 27 pages. https://doi.org/10.1145/3343172

[8] Jonny Whitmore, Thomas Seidl; *Shape Analysis: Automatically Detecting Formations*; Stats Perform; https://www.statsperform.com/resource/shape-analysis-automatically-detecting-formations/

[9] Jonathan Whitmore; *Introducing Movement Chain*; *Stats Perform*; https://www.statsperform.com/resource/introducing-movement-chains/.

[10] Rein, R., Memmert, D. Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science. SpringerPlus 5, 1410 (2016). https://doi.org/10.1186/s40064-016-3108-2

[11] Schumaker, Robert P. et al. "Sports Data Mining." (2010).

[12] Taki, Tsuyoshi and Jun-ichi Hasegawa. "Visualization of dominant region in team games and its application to teamwork analysis." Proceedings Computer Graphics International 2000 (2000): 227-235.

[13] Narizuka, T., Yamazaki, Y. (2019). Clustering algorithm for formations in football games. Scientific Reports, 9.