BUSINESS INTELLIGENCE INCLUDES FOOTBALL ANALYSIS

Boussoukaya Islam

Apr 20, 2022

CONTENTS

1 Project Context **3**
- 1.1 Real analytics . . . 3
- 1.2 Methodology . . . 4
- 1.3 Project objectives . . . 6
- 1.4 Data . . . 6

2 Kloppy Featuring InStat **7**
- 2.1 Introduction . . . 7
- 2.2 Problematic . . . 7
- 2.3 The Kloppy Package . . . 7
- 2.4 The InStat Provider . . . 12
- 2.5 Workflow . . . 15
- 2.6 Result . . . 21

3 Physical report **23**
- 3.1 Introduction . . . 23
- 3.2 Tracking Data . . . 23
- 3.3 Workflow . . . 24
- 3.4 Result . . . 26

4 Data Visualization **27**
- 4.1 Introduction . . . 27
- 4.2 FrameWork . . . 27
- 4.3 Dashboard Content . . . 28

5 Conclusion **33**

- *Project Context*
- *Kloppy Featuring InStat*
- *Physical report*
- *Data Visualization*
- *Conclusion*

CHAPTER

ONE

PROJECT CONTEXT

The use of data in sport, particularly in football, plays an important role in the development of players' skills or in the analysis of the game. Nowadays, we can talk about football in a more detailed way than before. This analysis could help clubs, managers or even individuals to identify and uncover new facts about football, this leads to an innovative and winning startegy.

The knowledge gained in my university years helped me achieve this project. The Business intelligence (BI) combines data analytics, data visualization, data mining, data tools and infrastructure. This project includes the first two terms, data analytics and data visualization on football.

1.1 Real analytics

1.1.1 Definition

Real Analytics is leading AI company of professors in mathematics, Statistics, Computer Science and economics. It has built a word-leading research in sports analytics. In 2018 Real Analytics was formed with the objective of using the research expertise in the football industry to mitigate the risk and uncertainty associated with player recruitment and team selection.

1.1.2 Achivements

- In january 2017, Real Analytics predicted Chelsea to win by seven points and Arsenal to finish outside the top 4.
- During the extraordinary 2016-17 Premier League season, they identified earlier than anyone that Leicester City were favourites for the league title. Their player ratings models predicted that Leicester City had a team capable of continuing their incredible start to the season.
- In April 2020 Real Analytics took a closer look at the impact of the potential sale of Pierre-Emerick Aubameyang for Arsenal FC. Their AI powered prediction models simulate the season more than 100k iteration under two scenarios: Arsenal keep Aubameyang, and Arsenal sell Aubamyang to Barcelona (Squad after). The results show that although there is no impact on Arsenal's chances of winning the league which stays at zero.

1.2 Methodology

Real Analytics chooses to treat scrum methodology as its core project management framework for the team to generate innovative and adaptive solutions for real-world problems. Adapting the Srcum framework to a company which has a small team with a heavy workload Scrum methodology will be the most appropriate choice.

1.2.1 What is SCRUM methodology ?

Scrum is an Agile project management methodology involving a small team led by a Scrum master, whose main job is to remove all obstacles to getting work done. Work is done in short cycles called sprints, and the team meets daily to discuss current tasks and any roadblocks that need clearing.

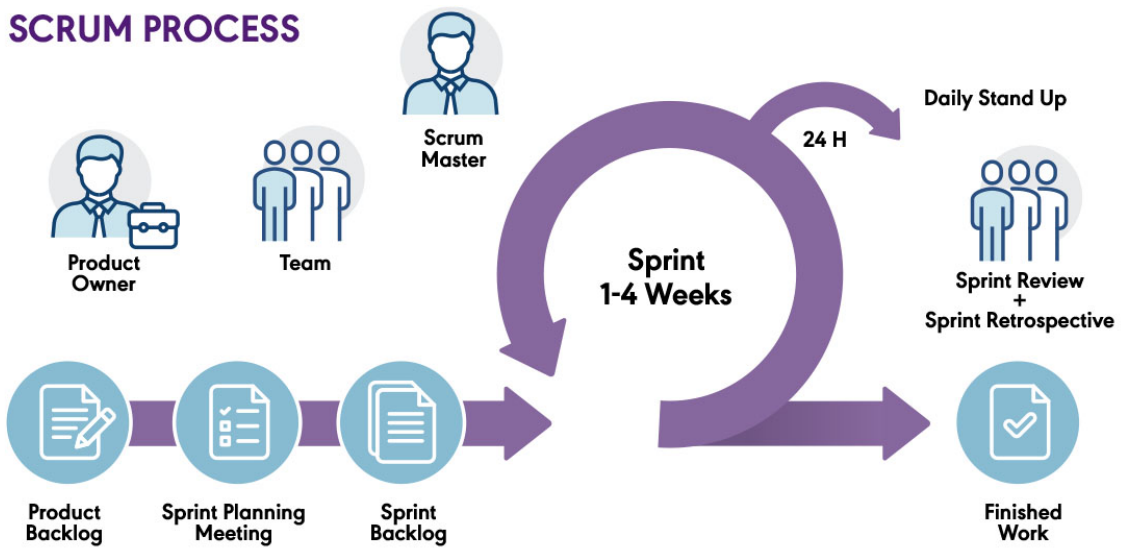

Fig. 1.1: SCRUM framework.

There are fundamentally 3 Scrum roles:

- **Product Owner** : The PO is a vital link between the development team and the product stakeholders, including the customer.
- **Scrum Master**: It is the role of the Scrum master to keep on top of the project's status and ensure that the development team follows the Scrum process.
- **Scrum Development Team**: The team is made up of 5-9 developers, programmers, designers and testers who collaborate. They do not have designated roles, but are given a specific set of tasks to complete in a given sprint.

Each sprint is subdivided into 5 phases :

- **Sprint planning**: During the sprint planning session, the Srum Master meets with the development team to plan details of the upcoming sprint. The team collectively decides which high-priority items in the product backlog can reasonably be completed during the sprint, given their available time and resources. Tasks are then assigned to individual team members.
- **Daily Scrum**: This brief daily is meeting limited to 15 minutes or less. The purpose of the daily Scrum is to check in with other team members, assess the day's progress, discuss any issues and plan the next day's work.

- **Sprint review**: The sprint review is a tune-up session that typically takes place on the last day of a sprint. It looks at what went well, what went wrong and what can be done better.
- **Sprint retrospective**: The retrospective is the final meeting at the end of a sprint, attended by the Product Owner, Scrum Master and development team. During the meeting, participants discuss what improvements can be made, and how to implement them in future sprints.
- **Sprint**: The sprint itself is considered an element of the development process. It encompasses all the work and events that occurred during its limited time increment.

1.2.2 Project

This project consists of 3 sprints, each sprint is a chapter.

- Sprint 1 : the kloppy package.
- Sprint 2 : Physical report.
- Sprint 3 : Visualization.

1.2.3 Tools

The company uses the following tools.

- **Jira Board**: Jira Board displays problems from one or more projects, giving a flexible way to see, manage and report on work in progress.

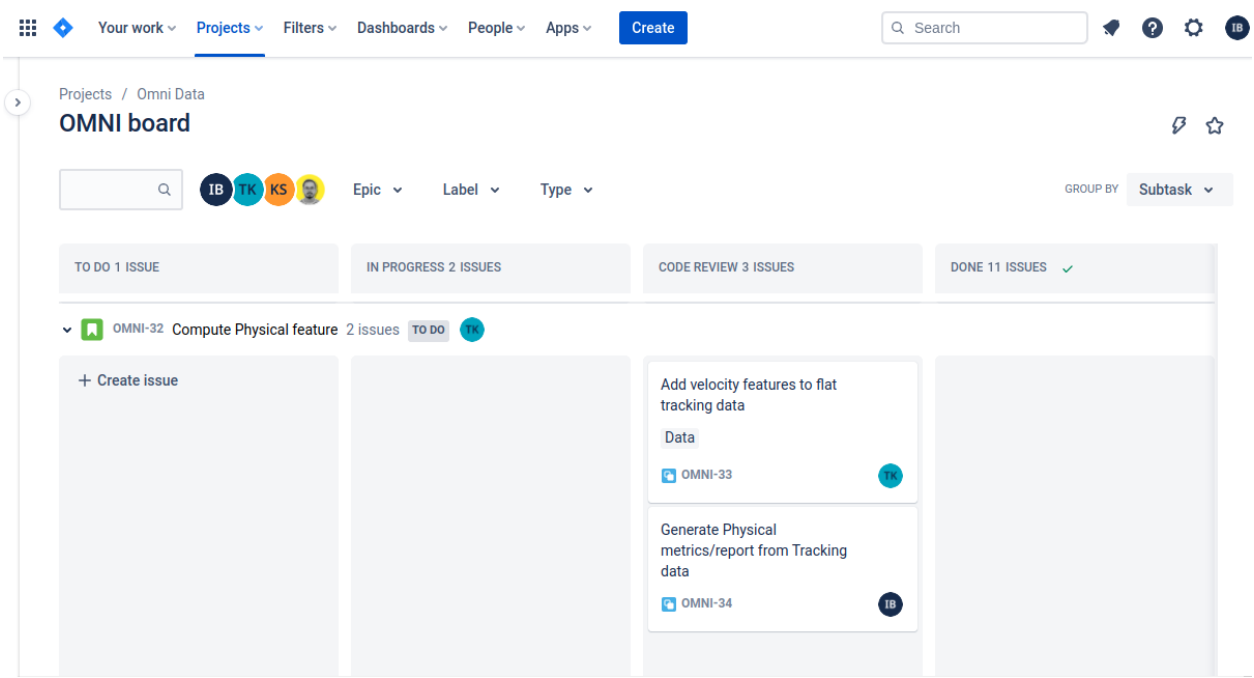

Fig. 1.2: Jira Board .

- **Slack**: Slack is a messaging app for business. It facilitates communication in the workplace. Helps teams communicate, collaborate across remote distances. Enables and encourages the sharing of documents, images and other assets across teams.
- **Google Meet**: Google meet serves for the daily virtual meetings.

1.3 Project objectives

The aim of this project is to build a tool to visualize tracking and event data of football games. The project consists of two main parts that will be divided into three independent chapters.

- **Processing Data** : Transform the raw data of the **Event data** from the data provider into a more user-friendly format and provide a standardized data model and Extract the player's physical summary report from the **Tracking data**.
- **Visualization** : Build a package to visualize the tracking data processed for a given football game (game score, players performance, player comparison).

1.4 Data

The data on which we will work on is created by remote sensing devices worn by prospective players and data gathered during actual gameplay, and supplied by dedicated data providers. The data we will be working on are tracking and event data that we intend to identify in the next chapters.

CHAPTER

TWO

KLOPPY FEATURING INSTAT

2.1 Introduction

Each football data provider uses its own unique format to describe the sequence of a game. Therefore, the software written to analyze this data must be suitable for a specific provider and cannot be used without modifications to analyze the data of other providers.

Kloppy is a Python package that addresses the challenges posed by the variety of data formats and aims to be the fundamental building block for processing soccer tracking and event data.

- How can the kloppy package deal with all these providers and generate the same result for each one? and which providers can support?

2.2 Problematic

Before delving into the details we will identify the problematic . The Company is delivered with football data from different providers, In our case, we will point to the Instat provider. We simply want to adapt the Instat provider to the Kloppy package to be able to use its many features.

2.3 The Kloppy Package

2.3.1 What is Kloppy ?

As previously mentioned kloppy is a great package , It provides (de)serializers, standardized data models, filters, and transformers to facilitate work with different tracking and event data from various providers,So its main features are to load data and transform it into standardized models and vice versa. In the following sections we will try to break-down the package to better understand the process and it works.

2.3.2 Kloppy providers

Kloppy support numerous provider with different format file (JSON & LXML) as the table below shows. To simplify the process, these providers display a different data structure within each format, "Kloppy" will proceed with each one to provide the same final result structure for each game.

Provider	Format
Datafactory	JSON
Metrica	JSON
Opta	LXML
Sportec	LXML
Statsbomb	JSON
Wyscout	JSON

2.3.3 What is events & line-up data in football?

- Events DataEvent data can be considered as a record of the entire game, it records every move on the pitch during the match. It includes the positions of each player and data pertaining to every action during the match which include pass, shot, interception, card, goal etc.
- Line-up Data The line-up data contain the lineup of players, starting & ending tactic . It includes player information full name , jersey number , starting & ending position etc.

2.3.4 Example

To better understand what we are looking for as a result here an code example and its output.

```python
from kloppy import opta

dataset = opta.load(
    f7_data="../../kloppy/tests/files/opta_f7.xml",
    f24_data="../../kloppy/tests/files/opta_f24.xml",

    # Optional arguments
    coordinates="opta",
    event_types=["pass", "shot"]
)

dataset.to_pandas().head()
```

	event_id	event_type	result	success	period_id	timestamp	end_timestamp	ball_state	ball_owning_team	team_id	player_id	coordinates_x	coo
0	2370983921	PASS	COMPLETE	True	1	0.160	None	alive	43	43	165809	50.1	
1	2370983925	PASS	COMPLETE	True	1	2.842	None	alive	43	43	146941	29.2	
2	2370983937	PASS	COMPLETE	True	1	5.019	None	alive	43	43	171314	34.7	
3	2370983945	PASS	COMPLETE	True	1	7.444	None	alive	43	43	121145	39.2	
4	2370983957	PASS	COMPLETE	True	1	9.322	None	alive	43	43	171314	24.0	

Fig. 2.1: Code exemple of Opta provider. Using filter for event types as pass and shot.

2.3.5 Kloppy Event Types

"Kloppy" classifies events depending on the type of event. For each event type kloppy defines a class that we will detail.

Events Types
Pass
Shot
Take on
Carry
Substitution
PlayerOn/Off
Card
Recovery
Ball Out
Foul Committed
Generic

Pass Event

A class entitled "Pass Event" contains the following attributes:

- recieve timestamp
- receiver player
- receiver coordinates **Point**
- result **Pass Result**

Attribute	Definition
COMPLETE	Complete pass
INCOMPLETE	Incomplete pass (intercepted)
OUT	Ball went out
OFFSIDE	Offside

- Pass type, a enumeration class named "Pass Type"

PassType
Cross
Hand pass
Head pass
High pass
Launch
Simple pass
Smart pass
Long pass
Through ball
Chipped pass
Flick on
Assit
Assist 2nd
Swithc of play

Shot Event

A class entitled “Shot Event” contains the following attributes:

- result of the shot as **shot Result**

Attribute	Definition
Goal	Shot resulted in a goal
Ooff targer	Shot was off target
Post	Shot hit the post
Blocked	Shot was blocked by another player
Saved	Shot was saved by the keeper

- result coordinates **Point**

Take on Event

A class entitled “Take-on Event” contains the following attributes:

- result of the take on **Take on Result**

Attribute	Definition
Complete	Complete take-on
Incomplete	Incomplete take-on
Out	Ball went out

Carry Event

A class entitled “Carry Event” contains the following attributes:

- end timestamp
- end coordinates **Point**
- Result **Carry Result**

Attribute	Definition
Complete	Complete carry
Incomplete	Incomplete carry

Substitution Event

A class entitled “Substitution Event” contains as attribute:

- replacement player **Player**

Player On/Off Event

Two classes "PlayerOn Event" & "PlayerOff Event" that contains as attribute:

- Event type that the player is concerned on/off

Card Event

A class named "Card Event", it contains as attribute:

- card type **Card Type**

Attribute	Definition
FIRST_YELLOW	First yellow card
SECOND_YELLOW	Second yellow card
RED	Red card

Recovery Event

A class named "Recovery Event", it contains as attribute:

- Event type **Recovery**

Ball out Event

A class named "Ball out Event", it contains as attribute:

- Event type **Ball out**

Foul Commit Event

A class named "Foul Commit Event", it contains as attribute:

- Event type **Foul Commit**

Generic Event

Kloppy classifies an event which does not match the above mentioned events (Unrecognised event type) as a generic event.

2.3.6 Kloppy qualifiers

Each type of event will be qualified with one or more of the following qualifiers.

Qualifiers	
Set Piece	Corner / FreeKick / Penalty /Throw in /KickOff
Body Part	Chest / Right Foot / Left Foot / Head
Pass Type	Cross / LongBall /ThroughBall / Launch / ChippedBall / Assist / 2nd Assist /SwitchOfPlay

2.3.7 Dependencies

Python libraries that Kloppy depend on

Black : Black is the uncompromising Python code formatter. By using it, you agree to cede control over minutiae of hand-formatting. In return, Black gives you speed, determinism, and freedom from pycodestyle nagging about formatting. Black makes code review faster by producing the smallest diffs possible.

Lxml : The lxml XML toolkit is a Pythonic binding for the C libraries libxml2 and libxslt. It is unique in that it combines the speed and XML feature completeness of these libraries with the simplicity of a native Python API, mostly compatible but superior to the well-known ElementTree API.

Requests : Python module that you can use to send all kinds of HTTP requests. It is an easy-to-use library with a lot of features ranging from passing parameters in URLs to sending custom headers and SSL Verification.

NetworkX : This python package is used for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. It is used to study large complex networks represented in form of graphs with nodes and edges. Using networkx we can load and store complex networks.

PyTest : PyTest is a testing framework that allows users to write test codes using Python programming language. It helps you to write simple and scalable test cases for databases, APIs, or UI. PyTest is mainly used for writing tests for APIs. It helps to write tests from simple unit tests to complex functional tests.

Pandas : pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

Pre-commit : A framework for managing and maintaining multi-language pre-commit Git hooks.

2.4 The InStat Provider

2.4.1 What is InStat ?

InStat is a worldwide sports performance analysis company, it prepares statistics for pre-game analysis, collects data during games and breaks down the game after it is completed.

2.4.2 InStat Files

As we mentionned before we will analyze the line-up & events file.

Line-up

Events

Each row is considered an event that occurred within the match, every action has the following characteristics: time, type of action , place on the pitch, the player who committed it, the opponent (in the case of challenges).

```
▼<data>
  <match id="1725171"/>
 ▼<first_team id="247" name="Liverpool FC" color="0xFF7788" color_num="0x000000">
  ▼<lineup>
   ▼<main starting_tactic="4-3-3 down" ending_tactic="4-3-3 down">
     <player id="3668" firstname="James Philip" lastname="Milner" num="7" starting_position_id="100"
     starting_position_name="Substitute player" ending_position_id="12" ending_position_name="Defender - Left"
     birthday="1986-01-04" country1_id="252" country1_name="England" starting_lineup="0" ending_lineup="1"/>
     <player id="3749" firstname="Xherdan" lastname="Shaqiri" num="23" starting_position_id="100"
     starting_position_name="Substitute player" ending_position_id="100" ending_position_name="Substitute player"
     birthday="1991-10-10" country1_id="233" country1_name="Switzerland" country2_id="96" country2_name="Kosovo"
     starting_lineup="0" ending_lineup="0"/>
     <player id="4322" firstname="Georginio Gregion Emile" lastname="Wijnaldum" num="5" starting_position_id="24"
     starting_position_name="Midfielder - Left central" ending_position_id="24" ending_position_name="Midfielder -
     Left central" birthday="1990-11-11" country1_id="138" country1_name="Netherlands" country2_id="198"
     country2_name="Suriname" starting_lineup="1" ending_lineup="0"/>
     <player id="5242" firstname="Thiago" lastname="Alcantara do Nascimento" num="6" starting_position_id="44"
     starting_position_name="Midfielder - Right central" ending_position_id="44" ending_position_name="Midfielder -
     Right central" birthday="1991-04-11" country1_id="77" country1_name="Spain" country2_id="29"
     country2_name="Brazil" starting_lineup="1" ending_lineup="1"/>
     <player id="5886" firstname="Jordan Brian" lastname="Henderson" num="14" starting_position_id="100"
     starting_position_name="Substitute player" ending_position_id="100" ending_position_name="Substitute player"
     birthday="1990-06-17" country1_id="252" country1_name="England" starting_lineup="0" ending_lineup="0"/>
     <player id="9403" firstname="Roberto Firmino" lastname="Firmino Barbosa De Oliveira" num="9"
     starting_position_id="36" starting_position_name="Forward - Central" ending_position_id="36"
     ending_position_name="Forward - Central" birthday="1991-10-02" country1_id="29" country1_name="Brazil"
     starting_lineup="1" ending_lineup="0"/>
     <player id="16425" firstname="Mohamed" lastname="Salah" num="11" starting_position_id="55"
     starting_position_name="Attacking midfielder - Right" ending_position_id="55" ending_position_name="Attacking
     midfielder - Right" birthday="1992-06-15" country1_id="65" country1_name="Egypt" starting_lineup="1"
     ending_lineup="1"/>
```


Fig. 2.2: This a sample of the lineup XML file


```
<match id="1725171"/>
<row id="5137" number="1" action_id="16310" action_name="GK" player_id="137016" player_name="Alisson Ramses Alisson"
team_id="247" team_name="Liverpool FC" position_id="31" position_name="Goalkeeper" half="1" second="0"
standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5139" number="66" action_id="16520" action_name="RD" player_id="272150" player_name="Trent Alexander-Arnold"
team_id="247" team_name="Liverpool FC" position_id="52" position_name="Defender - Right" half="1" second="0"
standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5141" number="47" action_id="16420" action_name="RCD" player_id="424786" player_name="Nathaniel Phillips"
team_id="247" team_name="Liverpool FC" position_id="42" position_name="Defender - Right central" half="1" second="0"
standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5143" number="46" action_id="16220" action_name="LCD" player_id="395318" player_name="Rhys Williams"
team_id="247" team_name="Liverpool FC" position_id="22" position_name="Defender - Left central" half="1" second="0"
standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5145" number="26" action_id="16120" action_name="LD" player_id="157248" player_name="Andrew Robertson"
team_id="247" team_name="Liverpool FC" position_id="12" position_name="Defender - Left" half="1" second="0"
standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5147" number="6" action_id="16440" action_name="RCM" player_id="5242" player_name="Thiago Alcantara do
Nascimento" team_id="247" team_name="Liverpool FC" position_id="44" position_name="Midfielder - Right central"
half="1" second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5149" number="5" action_id="16240" action_name="LCM" player_id="4322" player_name="Georginio Gregion Emile
Wijnaldum" team_id="247" team_name="Liverpool FC" position_id="24" position_name="Midfielder - Left central" half="1"
second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5151" number="3" action_id="16330" action_name="CDM" player_id="120900" player_name="Fabio Henrique Tavares"
team_id="247" team_name="Liverpool FC" position_id="33" position_name="Defensive midfielder - Central" half="1"
second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5153" number="10" action_id="16150" action_name="LAM" player_id="88763" player_name="Sadio Mane"
team_id="247" team_name="Liverpool FC" position_id="15" position_name="Attacking midfielder - Left" half="1"
second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5155" number="9" action_id="16360" action_name="CF" player_id="9403" player_name="Roberto Firmino Firmino
Barbosa De Oliveira" team_id="247" team_name="Liverpool FC" position_id="36" position_name="Forward - Central"
half="1" second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
<row id="5157" number="11" action_id="16550" action_name="RAM" player_id="16425" player_name="Mohamed Salah"
team_id="247" team_name="Liverpool FC" position_id="55" position_name="Attacking midfielder - Right" half="1"
second="0" standart_id="1" standart_name="Open play" ts="2021-05-24T00:41:57.365" dl="0"/>
```


Fig. 2.3: This a sample of the events XML file

2.4.3 InStat Actions

InStat has a specific way of structuring events as well as its type of action, each action has a specefic identifier .

GENERIC ACTION	ACTION
Pass	Attacking pass - Key pass - Assist - Key Assit
Challenge	Challenge - Air challenge - Tackle - Dribble
Disipline	Yellow card - Red card - RC for 2 YC
Shot	Shot on target - Shot into the bar - Shot blocked
Set Piece	Open play - Throw-in - Free-Kick - Corner
Goal	Goal - Own goal
Cross	Cross accurate - cross accurate

2.4.4 InStat Coordinates System

The Coordinates system is X,Y coordinates of the player on the pitch , it refers to his position. The origin of the coordinates is in the bottom left and the vertical orientation is from bottom to top of the pitch. The dimension of the picth is 105 * 68 .

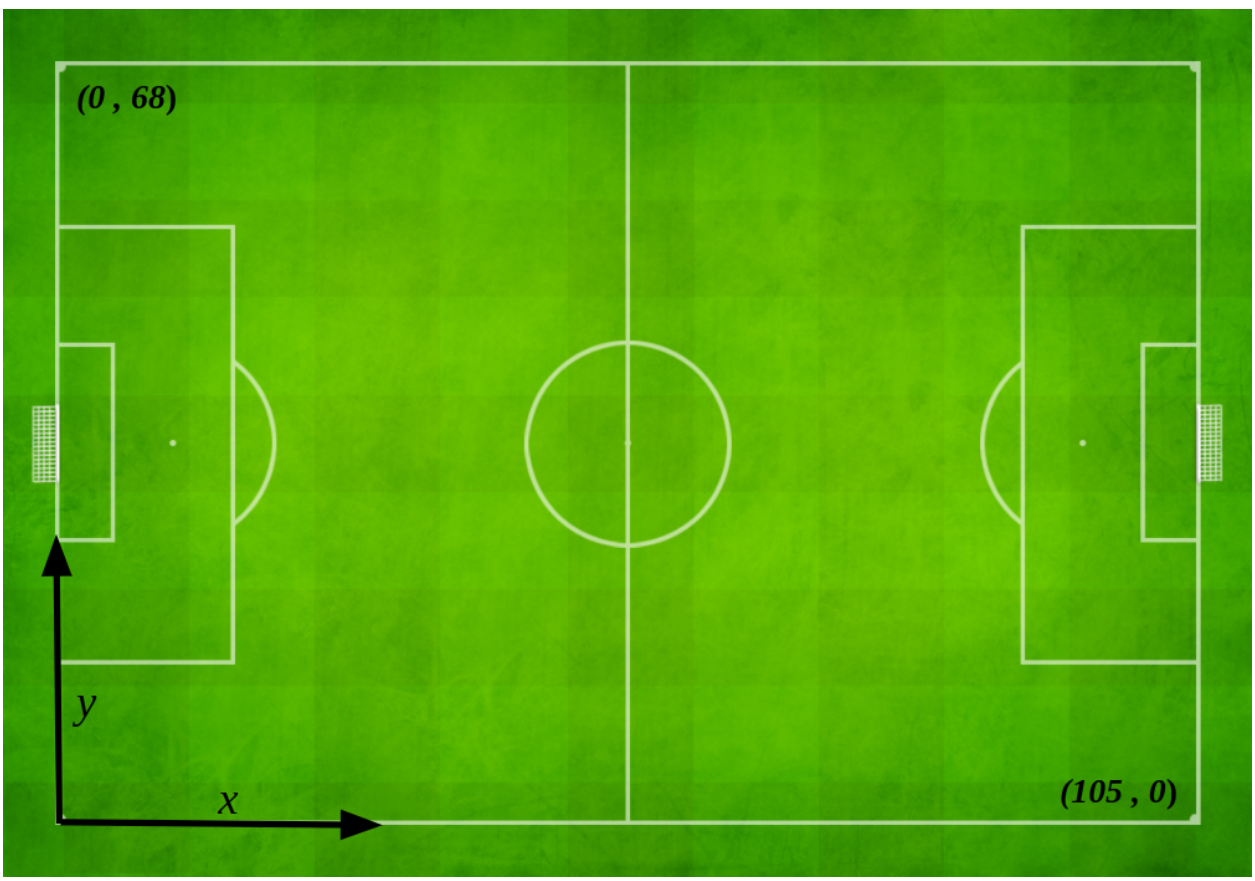

Fig. 2.4: InStat pitch representation.

2.4.5 InStat Standart

Every event is qualified with a standart name as the following table shows.

Standart name
Open Play
Throw-in
Indirect Freekick
Direct free kick
Corner

2.4.6 InStat Body Part

Every event is qualified with a Body name that reffers to the body part as the following table shows.

Body name
Right foot
Left foot
Header
Hand
Other body part

2.5 Workflow

In this section, we will describe in detail the approach adopted.

2.5.1 Instat Inputs

Every provider in Kloppy has an input class that contains the files that will be used. We created the InStatInputs class which has as attributes the file line-up and events as the following code states.

```
class InStatInputs(NamedTuple):
    lineup_data: IO[bytes]
    events_data: IO[bytes]
```

2.5.2 Coordinates System

Each provider has its own coordinate system, so we defined a data class that that specifies the pitch parameters.

```
@dataclass
class InstatCoordinateSystem(CoordinateSystem):
    @property
    def provider(self) -> Provider:
        return Provider.INSTAT

    @property
    def origin(self) -> Origin:
```

(continues on next page)

(continued from previous page)

```
        return Origin.BOTTOM_LEFT

    @property
    def vertical_orientation(self) -> VerticalOrientation:
        return VerticalOrientation.BOTTOM_TO_TOP

    @property
    def pitch_dimensions(self) -> PitchDimensions:
        return PitchDimensions(
            x_dim=Dimension(0, 105),
            y_dim=Dimension(0, 68),
        )
```

2.5.3 Action Id

In this section, we will try to match the types of InStat events and actions to Kloppy classes in order to proceed with the deserialization process. We have created a file called "InStat Events" that contains all types of events with their identifiers. This is a sample of the code.

```
EVENT_TYPE_BALL_OUT = "27000"
EVENT_TYPE_CORNER_AWARDED ="5060"
BALL_OUT_EVENTS = [EVENT_TYPE_BALL_OUT, EVENT_TYPE_CORNER_AWARDED]

EVENT_BODYPART_RIGHT_FOOT = 1
EVENT_BODYPART_LEFT_FOOT = 2
EVENT_BODYPART_HEADER = 3
EVENT_BODYPART_BODY = 4
EVENT_BODYPART_HAND = 5

SET_PIECE_THROW_IN = 2
SET_PIECE_INDIRECT_FREE_KICK = 3
SET_PIECE_DIRECT_FREE_KICK = 4
SET_PIECE_CORNER = 5
```

2.5.4 Qualifiers Functions

Qualifiers describe each single event with one or more of these following qualifiers (Body part, Card, Set piece and Pass).

Body Part Qualifier

This qualifier function returns which body part is used with each event.

```
def _get_event_bodypart(body_id : int) -> List[Qualifier]:
    qualifiers =  []
    if body_id == instat_events.EVENT_BODYPART_RIGHT_FOOT:
        qualifiers.append(BodyPartQualifier(value=BodyPart.RIGHT_FOOT))
    elif body_id == instat_events.EVENT_BODYPART_LEFT_FOOT:
        qualifiers.append(BodyPartQualifier(value=BodyPart.LEFT_FOOT))
    elif body_id == instat_events.EVENT_BODYPART_HEADER:
        qualifiers.append(BodyPartQualifier(value=BodyPart.HEAD))
```

(continues on next page)

(continued from previous page)

```
    elif body_id == instat_events.EVENT_BODYPART_BODY :
        qualifiers.append(BodyPartQualifier(value=BodyPart.OTHER))
    elif body_id == instat_events.EVENT_BODYPART_HAND :
        qualifiers.append(BodyPartQualifier(value=BodyPart.OTHER))

    return qualifiers
```

Card Qualifier

This qualifier function returns which card type is used with each card event.

```
def _get_event_card(action_id : str) -> List[Qualifier]:
    qualifiers =  []
    if action_id == instat_events.EVENT_TYPE_RED_CARD:
        qualifiers.append(CardQualifier(value=CardType.RED))
    elif action_id == instat_events.EVENT_TYPE_SECOND_YELLOW_CARD:
        qualifiers.append(CardQualifier(value=CardType.SECOND_YELLOW))
    elif action_id == instat_events.EVENT_TYPE_FIRST_YELLOW_CARD:
        qualifiers.append(CardQualifier(value=CardType.FIRST_YELLOW))
    return qualifiers
```

Pass Qualifier

This qualifier function returns the type of the pass concerned with each pass event.

```
def _get_event_pass(action_id : str) -> List[Qualifier]:
    qualifiers =  []
    if action_id in instat_events.EVENT_TYPE_CROSS:
        qualifiers.append(PassQualifier(value=PassType.CROSS))
    elif action_id in instat_events.EVENT_TYPE_ASSIST:
        qualifiers.append(PassQualifier(value=PassType.ASSIST))
    elif action_id in instat_events.EVENT_TYPE_ASSISIT_2ND:
        qualifiers.append(PassQualifier(value=PassType.ASSIST_2ND))
    return qualifiers
```

Set Piece Qualifier

As we mentioned previously, the equivalent of set piece in Instat is **standart**. This qualifier function returns the type of the set piece concerned with each event.

```
def _get_event_setpiece(standart_id) -> List[Qualifier]:
    qualifiers = []
    if standart_id == instat_events.SET_PIECE_CORNER:
        qualifiers.append(SetPieceQualifier(value=SetPieceType.CORNER_KICK))
    elif standart_id == instat_events.SET_PIECE_DIRECT_FREE_KICK:
        qualifiers.append(SetPieceQualifier(value=SetPieceType.FREE_KICK))
    elif standart_id == instat_events.SET_PIECE_INDIRECT_FREE_KICK:
        qualifiers.append(SetPieceQualifier(value=SetPieceType.FREE_KICK))
    elif standart_id == instat_events.SET_PIECE_THROW_IN:
        qualifiers.append(SetPieceQualifier(value=SetPieceType.THROW_IN))
    return qualifiers
```

2.5.5 Parse Functions

The parse functions are used to retrieve data from the file to specific classes set by kloppy. This is where de-serialization takes its parts and transforms the data extracted from the XML file into the classes identified by KLoppy. As stated in the preceding chapter, Qualifier describes each event type with the correspondents qualifiers.

Parse Team

This function identifies the team (id, name, ground, team side and starting formation) & players of the team (player id, team, jersey number, first name, last name, starting, position) as the following code shows.

```
def _parse_team(team_root , team_side
                )-> Team:
    team_id = team_root.attrib["id"]
    formation = "-".join(re.findall(r'\d+', team_root.lineup.main.attrib["starting_
↪tactic"]))
    team = Team(
        team_id=str(team_id),
        name=team_root.attrib["name"],
        ground=Ground.HOME
        if str(team_side) == "first_team"
        else Ground.AWAY,
        starting_formation=FormationType(formation),
    )
    team.players = [
        Player(
            player_id=player_elm.attrib["id"],
            team=team,
            jersey_no=int(player_elm.attrib["num"]),
            first_name=player_elm.attrib["firstname"],
            last_name=player_elm.attrib["lastname"],
            starting=True if player_elm.attrib["starting_lineup"] == 1 else False,
            position=Position(
                position_id=player_elm.attrib["starting_position_id"],
                name=player_elm.attrib["starting_position_name"],
                coordinates=None,
            ),
        )
        for player_elm in team_root.lineup.main.iterchildren("player")
    ]
    return team , team_id
```

Parse Score

This function returns the score of both teams, the home and away team.

```
def _parse_score (events_root,home_team_id,away_team_id):

    home_score = 0
    away_score = 0
    try:
        for event in events_root.iterchildren("row"):
            if event.attrib["action_id"]=="8010" and event.attrib["team_id
↪"]==str(home_team_id):
```

(continues on next page)

(continued from previous page)

```
                home_score +=1
            elif event.attrib["action_id"]=="8010" and event.attrib["team_id
↪"]==str(away_team_id):
                away_score +=1

    except KeyError:
        pass
    return home_score,away_score
```

Parse Card

This function returns the type of the card.

```
def _get_event_card(action_id : str) -> List[Qualifier]:
    qualifiers =  []
    if action_id == instat_events.EVENT_TYPE_RED_CARD:
        qualifiers.append(CardQualifier(value=CardType.RED))
    elif action_id == instat_events.EVENT_TYPE_SECOND_YELLOW_CARD:
        qualifiers.append(CardQualifier(value=CardType.SECOND_YELLOW))
    elif action_id == instat_events.EVENT_TYPE_FIRST_YELLOW_CARD:
        qualifiers.append(CardQualifier(value=CardType.FIRST_YELLOW))
    return qualifiers
```

Parse pass

This function takes as an input a pass event and the action id, it returns the attributes used in the Pass Event class.

```
def _parse_pass(action_id: str, row_elm) -> Dict:
    if action_id in instat_events.EVENT_TYPE_CROSS:
        if action_id in instat_events.EVENT_TYPE_CROSS_INCOMPLETE:
            result = PassResult.INCOMPLETE
        elif action_id in instat_events.EVENT_TYPE_CROSS_COMPLETE:
            result = PassResult.COMPLETE
    elif action_id in instat_events.EVENT_TYPE_ASSIST:
        result = PassResult.COMPLETE
    elif action_id in instat_events.EVENT_TYPE_ASSISIT_2ND:
        result = PassResult.COMPLETE

    receiver_coordinates = Point(
            x=float(row_elm.attrib["pos_dest_x"]), y=float(row_elm.attrib["pos_dest_y
↪"])
        )
    qualifiers = _get_event_qualifiers(row_elm)

    return dict(
        result=result,
        receiver_coordinates=receiver_coordinates,
        receiver_player=None,
        receive_timestamp=None,
        qualifiers=qualifiers,
    )
```

Parse shot

This function takes as an input a shot event, the action id, and the coordinates. It returns the attributes used in the Shot Event class.

```
def _parse_shot(
            action_id: str,
            coordinates: Point,
            row_elm
               ) -> Dict:
    if action_id == instat_events.EVENT_TYPE_SHOT_GOAL:
        result = ShotResult.GOAL
    elif action_id == instat_events.EVENT_TYPE_SHOT_OWN_GOAL:
        result = ShotResult.OWN_GOAL
    elif action_id == instat_events.EVENT_TYPE_SHOT_BLOCKED:
        result = ShotResult.BLOCKED
    elif action_id == instat_events.EVENT_TYPE_SHOT_POST:
        result = ShotResult.POST
    elif action_id == instat_events.EVENT_TYPE_SHOT_SAVED:
        result = ShotResult.SAVED
    else:
        result = None
    qualifiers = _get_event_qualifiers(row_elm)
    return dict(
        coordinates=coordinates,
        result=result,
        qualifiers=qualifiers)
```

Parse take-on

This function takes as an input the action id and it returns the result of the take on.

```
def _parse_take_on(action_id: str) -> Dict:
    if action_id in instat_events.EVENT_TYPE_TAKE_ON_COMPLETE:
        result = TakeOnResult.COMPLETE
    elif action_id == instat_events.EVENT_TYPE_TAKE_ON_INSUCC_DRIBBLING:
        result = TakeOnResult.INCOMPLETE
    return dict(result=result)
```

2.5.6 InStat Deserializer

It is the final class that includes all previous sections and will return the final Dataset after the de-serialization process took part.

```
return EventDataset(
            metadata=metadata,
            records=events,
        )
```

EventDataset is a data class that contains the metadata & records of the game.

Metadata

The Metadata contains both team (home & away team), Period (timestamp of the first & second half), Score of the game, coordiantes system of Instat .

```
metadata = Metadata(
            teams=teams,
            periods=periods,
            pitch_dimensions=transformer.get_to_coordinate_system().pitch_dimensions,
            score=score,
            frame_rate=None,
            orientation=Orientation.ACTION_EXECUTING_TEAM,
            flags=DatasetFlag.BALL_OWNING_TEAM,
            provider=Provider.INSTAT,
            coordinate_system=transformer.get_to_coordinate_system(),
        )
```

Records

The records refer to the list of events after it is generated (de-serialized) into Kloppy classes.

2.6 Result

```
dataset = instat.load(
    lineup_data="/home/islam/Downloads/data/lineup_1725171.xml",
    events_data="/home/islam/Downloads/data/events_1725171.xml",

    # Optional arguments
    coordinates="INSTAT",

)
dataset.to_pandas().head()
```

	event_id	event_type	result	success	period_id	timestamp	end_timestamp	ball_state	ball_owning_team	team_id	player_id	coordinates_x	coor
0	10074	GENERIC:unknown	None	None	1	0.00	None	alive	1014	1014	26667	52.0	
1	10075	GENERIC:unknown	None	None	1	1.67	None	alive	1014	1014	6000	37.0	
2	10076	GENERIC:unknown	None	None	1	4.79	None	alive	247	247	424786	25.5	
3	12733	GENERIC:unknown	None	None	1	4.79	None	alive	247	247	424786	25.5	
4	25581	RECOVERY	None	None	1	4.79	None	alive	247	247	424786	25.5	

Fig. 2.5: Code exemple

CHAPTER

THREE

PHYSICAL REPORT

3.1 Introduction

The evaluation of the players performance is mandatory step in the football analysis. It allows to analyze each profile, compare profiles. The player's physical report includes player performance, such as the total distance covered, the total number of minutes played, the number of sprints, the maximum speed, etc. In this chapter, we will work on the tracking data and how we will generate the player's physical report from it. Besides that we want to allow the user to configure the prameters of each type of running.

3.2 Tracking Data

The tracking data gives a detailed overview of each player's position and movement on the pitch at all times. It provides a vital context for everything that goes on on the pitch, on and off the ball.

	A	B	C	D	E	F	G
1	frameI	periodI	seconds	possessionSide	away_player_1_x	away_player_1_y	away_player_1_team_id
2	740	1	0	unknown	93.8	34.34	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
3	741	1	0.1	home	93.69	34.44	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
4	742	1	0.2	home	93.6	34.52	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
5	743	1	0.3	home	93.53	34.57	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
6	744	1	0.4	home	93.46	34.6	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
7	745	1	0.5	home	93.4	34.57	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
8	746	1	0.6	home	93.37	34.57	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
9	747	1	0.7	home	93.36	34.57	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
10	748	1	0.8	home	93.35	34.56	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
11	749	1	0.9	home	93.34	34.54	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
12	750	1	1	home	93.32	34.51	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
13	751	1	1.1	home	93.31	34.47	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
14	752	1	1.2	home	93.29	34.44	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
15	753	1	1.3	home	93.28	34.42	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
16	754	1	1.4	home	93.28	34.4	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
17	755	1	1.5	home	93.28	34.38	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
18	756	1	1.6	home	93.28	34.36	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
19	757	1	1.7	home	93.29	34.33	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
20	758	1	1.8	home	93.31	34.29	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
21	759	1	1.9	home	93.34	34.26	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
22	760	1	2	home	93.38	34.22	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d
23	761	1	2.1	home	93.41	34.19	66fae705a10eba94c95996e92357a5a6e4988f27152e19905d

Fig. 3.1: This is a sample of tracking data.

Each frame in the above figure includes the period (1 if the first half and 2 if the second half),The side team of he player (home or away team), The position (value of X and Y of the pitch) for each player, a boolean variable whether the player is visualised or not and the recorded speed.

3.3 Workflow

The player's physical report includes these following variables :

- **Total Minutes**: Total minutes in MM:SS format that a player played in the game.
- **Distance**: Total distance player travelled in meters.
- **Standing**: Total distance player travelled in meters while standing. Player is defined as standing if they are moving <= 0.2 m/s.
- **Walking**: Total distance player travelled in meters while walking. Player is defined as walking if they are moving >0.2 m/s and <= 2.0 m/s.
- **Jogging**: Total distance player travelled in meters while jogging. Player is defined as jogging if they are moving >2.0 m/s and <= 4.0 m/s.
- **Running**: Total distance player travelled in meters while running. Player is defined as running if they are moving >4.0 m/s and <= 5.5 m/s.
- **High Speed Running**: Total distance player travelled in meters while high speed running.Player is defined as high speed running if they are moving >5.5 m/s and <= 7.0 m/s.
- Sprint: Total distance player travelled in meters while sprinting. Player is defined as sprinting if they are moving >7.0 m/s.
- **Top Speed**: The peak velocity that the player reached in the entire game. Top speed is reported in km/h.
- **No. of High Intensity Runs**: The number of high speed runs and sprints that the player performs in a game.

3.3.1 Minutes played

To compute the total minutes played we have to :

- Identify the first and the last frame that the player appears on.
- Calculate the difference in seconds. If the last observation is in the second half, and the first in the first half, the total length of the first half should be considered.

	A	B	C	D	J	K
1	frameId	periodId	seconds	possessionSide	away_player_4_x	away_player_4_y
2	740	1	0	unknown	. 72.01	40.49

	A	B	C	D	J	K
36109	59623	2	2044.9	home	50.68	28.89

Fig. 3.2: Exemple of the first and last frame of away player 4.

```
In [ ]:
_played_minutes(
    "away_player_4_player_id",
    tracking_df,
    tracking_df[tracking_df.periodId == 1].seconds.max(),
)

Out[ ]: 97.63000000000001
```

Fig. 3.3: Code exemple.

3.3.2 Distance covered

for each run type we will compute the distance covered, we must go through these steps:

- compute the sampling freqency, it is the median time in seconds between frames.
- define the running window. In our case, it is the number of consecutive frames in one minute period.

	A	B	C	D
1	frameId	periodId	seconds	possessionSide
2	0	1	0	home
3	1	1	0.04	home
4	2	1	0.08	home
5	3	1	0.12	home
6	4	1	0.16	home
7	5	1	0.2	home
8	6	1	0.24	home
9	7	1	0.28	home
10	8	1	0.32	home
11	9	1	0.36	home
12	10	1	0.4	home
13	11	1	0.44	home
14	12	1	0.48	home
15	13	1	0.52	home
16	14	1	0.56	home
17	15	1	0.6	home
18	16	1	0.64	home
19	17	1	0.68	home
20	18	1	0.72	home
21	19	1	0.76	home
22	20	1	0.8	home
23	21	1	0.84	home
24	22	1	0.88	home
25	23	1	0.92	home
26	24	1	0.96	home
27	25	1	1	home

	A	B	C	D
1	frameId	periodId	seconds	possessionSide
2	740	1	0	unknown
3	741	1	0.1	home
4	742	1	0.2	home
5	743	1	0.3	home
6	744	1	0.4	home
7	745	1	0.5	home
8	746	1	0.6	home
9	747	1	0.7	home
10	748	1	0.8	home
11	749	1	0.9	home
12	750	1	1	home
13	751	1	1.1	home
14	752	1	1.2	home
15	753	1	1.3	home
16	754	1	1.4	home
17	755	1	1.5	home
18	756	1	1.6	home
19	757	1	1.7	home
20	758	1	1.8	home
21	759	1	1.9	home
22	760	1	2	home
23	761	1	2.1	home
24	762	1	2.2	home
25	763	1	2.3	home
26	764	1	2.4	home
27	765	1	2.5	home

Fig. 3.4: the number of running window of the left provider is 25 and for the right provider is 10.

- For each frame of the corresponding player we verify the condition of each run type and multiply this by the sampling frequency. Finally, we sum up these frames.

And to compute the number of high speed running, we have to verify it's condition (>5.5 m/s and <= 7.0 m/s) and find the number of occassions it was sustained for at least one running window length.

3.3.3 Top speed

To determine the top speed of the player, we look for the maximum speed recorded.

3.3.4 Exemple

In [ ]:
```
run_info = _running_info("away_player_4_player_id", tracking_df, 0.1)
run_info
```

Out[ ]:

Distance	Standing	Walking	Jogging	Running	High-Speed Running	Sprinting	# High Intensity Runs	# Sprints	playerId	Top Speed
8094.64712	14.074041	2728.782621	3640.448663	1178.833892	430.374153	102.13375	32	6	27a873ab8064d003025d1809775a404ea6d66f3ce24ea8...	31.087181

Fig. 3.5: Exemple of the away player 4.

3.4 Result

	A	B	C	D	E	F	G	H	I	J
1	Distance	Standing	Walking	Jogging	Running	High-Speed Running	Sprinting	# High Intensity Runs	# Sprints	playerId
2	10694.1718065	10.43084029195	2046.76578603	4588.93223034	2901.90782373	1010.62230210522	135.5128239893	61	4	cedfe37314f4cda689006461498a507b1b33009e076f5cf44
3	9889.40079261	8.541925435888	2418.57466171	4786.49196874	2031.47073915	486.127581602414	158.1939159658	37	7	aa96464d2088f91d0f9fe2e78e6d609854dc28860f10cc1f9
4	9636.60273568	8.271827579314	2565.6676437	3997.07589207	1748.27119568	791.868908107166	525.4472685385	48	21	3536a7999c8445b5e976a97ac29ddaeb67f8617ff3b8d73b
5	9632.39239159	7.91113167711	2432.18004544	4716.39001356	1719.65652826	508.255454238654	247.9992184104	29	13	1d947c22bc5f4a732fb3f3dfb07d5a4f9715d423bf981a7a5e
6	9452.84777503	9.592326257213	2410.3376688	4719.63806551	1701.51146004	501.127315864257	110.6409385593	36	7	efe57814ad66d98aec04cec9279350a9454065bebacac0a
7	9413.42527395	7.665213192602	2493.84379633	4364.04781238	1665.55242927	597.152946569956	285.1630762083	36	13	4fdd6d44f27fccd6926b99b66b96ac31acb58d78aa3c5a38
8	9369.29054232	13.72483362233	2434.38899383	4132.13430513	1845.10583901	684.398677956073	259.5378927794	49	10	433ac3bb16333f5538cae2c0874dfa5042e62951b43237b
9	9347.70268698	9.381310356037	2731.08290354	3803.16388074	1653.20304623	831.808330593631	319.0632155295	57	18	6789255fa015dd9f081e05d224552ca72e61afd17309ed6b
10	9259.48979419	8.875365954376	2662.34060293	3926.6728948	1595.437267	712.710354097566	353.4533094116	42	13	eed08a1e908beb8c77510b8d8ee0dab681d611fe5a77161
11	9221.36760715	9.560428765059	1779.88944279	4502.51371031	2486.06294615	426.560865948741	16.7802131818	30	0	a093f6b55254780919cddeec4b607b4b72b97497f7acfa4a
12	9106.91402346	11.77923281888	2658.63692373	4052.87495987	1472.02786995	622.449768217266	289.1452688644	39	10	fb80b18384aa51fa7275361f44101528774e988c3aee3497
13	9062.37977414	10.95641221217	2549.94769156	4163.97234961	1701.03070767	567.045176242143	69.42743684438	37	2	04583a17d5b509c171892bcbf4d3a5b9ed2dfa519c06b432
14	8872.95610442	6.342270456185	1692.26204214	4220.84032579	2204.04126794	711.872256926026	37.59794116607	40	1	2f40e3a0e62efcd2b4e66489453a7dd1a0162d1c48435e38
15	8744.44238397	6.058345130571	2029.4535488	3694.43778601	1926.62012884	825.924606771094	261.9479684175	50	12	d852020e305114abbdef2c6680626fd9743629a930f19471
16	8667.25485567	13.39959199634	2623.34506432	3952.91988536	1360.57751618	443.369176763249	273.643621043	26	14	a295a138484cecf22212ad9f65be919fd385667e19fc88494
17	8197.08697966	11.73823085455	2824.00885942	3773.61378987	1086.16335554	343.17550199592	158.3872419747	20	6	957e96878335293e1b2d97a906ccf182c7253902141584f4e
18	8094.64712011	14.0740411578	2728.78262121	3640.44866254	1178.83389199	430.374152747868	102.133750462	32	6	27a873ab8064d003025d1809775a404ea6d66f3ce24ea89
19	7437.79827452	9.097315741481	2226.23604804	3407.2565664	1094.05260964	503.318116502146	197.8376181905	33	10	9f43f887be46814fd9e7190b0296432ddd2c83ea3ac1aabfc
20	6361.05020954	5.613099389927	1383.69805138	3301.00661062	1233.23580141	378.665958528979	58.83068820623	23	2	959d1756675b93dfda464e3c7c5edf58f038f0608006b4072
21	4690.78280538	7.444587341696	1525.17566534	1887.1540143	689.872977864	367.251799431257	213.8837611127	24	13	56f5838ecac19132f1c7db958670c8467dcf94d9d0d076eb5
22	4556.81803387	6.655409677239	935.123419115	2158.8155975	966.551298586	368.138245442749	121.5340635449	27	6	f53baced004e706c14e75f1517bd2ed17bae041919fa9c9b
23	3839.22189272	42.79436230664	2895.3045852	750.638157688	95.5154450839	46.0269486197886	8.942393821802	3	0	587be382e0c2d39f10ae637a2667d423288e199e6055bcb
24	3590.08883166	2.18460939595	913.912752102	1528.35437214	817.245095354	262.685938291076	65.70606437453	16	2	929239479424b9cfab160960d670d5115a4cf9387b8104c8
25	3284.62742432	53.15195645219	2897.91000309	321.358028218	12.2074365619	0	0	0	0	d8db35953e75ba6c25c946c1fadea2dca90114fccbe767f1c
26	1810.72943852	1.027512237361	296.249093523	864.980996132	464.145677981	142.991005973854	41.33515267136	9	1	71ef9f1c4c2b2fa5c67577991995e191f7d7189f3ff6863c392
27	1610.67352941	1.793840168343	407.14927297	619.244942885	416.032427796	139.572210819539	26.88083476842	10	2	c9ab8bb7ae658236ce767bd91549698ac426233f85e3f680
28	1453.56391535	1.224914785315	426.82902886	593.969156133	270.539935844	140.484351684014	20.516528043	10	1	8657aa43eb6230e347e7065ee35e165040781a8e36625e
29	1170.8058096	0.585225382059	289.192806913	541.036629402	170.232989455	108.908333334649	60.84982511904	10	4	5fd2a9618f9c3e52194d4fd115c9177d8e5007b13f423e9c1

Fig. 3.6: The physical report.

CHAPTER

FOUR

DATA VISUALIZATION

4.1 Introduction

In this chapter, we will visualize the data we were working on, so that we can present them with the best layout and the best way.

4.2 FrameWork

Away from the basic softwares used for data visualisation, We will stick to Python and create the dashboard from scratch using dash-plotly as a framework. We're going to build an analytical web application.

4.2.1 What is Dash ?

Dash is a python framework created by plotly for creating interactive web applications. Dash is written on the top of Flask, Plotly.js and React.js.Dash is open source and the application build using this framework are viewed on the web browser.

The Dash application is made of two building blocks

Layout : It defines the structure of the application . Elements used such as dropdowns , buttons , graphs etc and the placement , size , color etc. Dash contains Dash HTML components which we can create and style HTML content such as heading, paragraph, images etc using python . Elements such as dropdowns, graphs, cards are created using Dash Core components. Dash also includes Dash Bootstrap components that makes it easier to build consistently styled apps with complex, responsive layouts.

Callbacks : Are used to bring interactivity to the dash application. These are the functions using which, for example, we can define the activity that would happen on clicking a button or a dropdown.

4.2.2 Dash Bootstrap Components

For improved layout design, we will use Bootstrap. Dash-bootstrap-components is a library of Bootstrap components for Plotly Dash, that makes it easier to build consistently styled apps with complex, responsive layouts.

4.3 Dashboard Content

4.3.1 Match page

The landing page which contains game information such as home team and away team, name and score also two buttons to pass to display the line-up for each team and comparison of players.

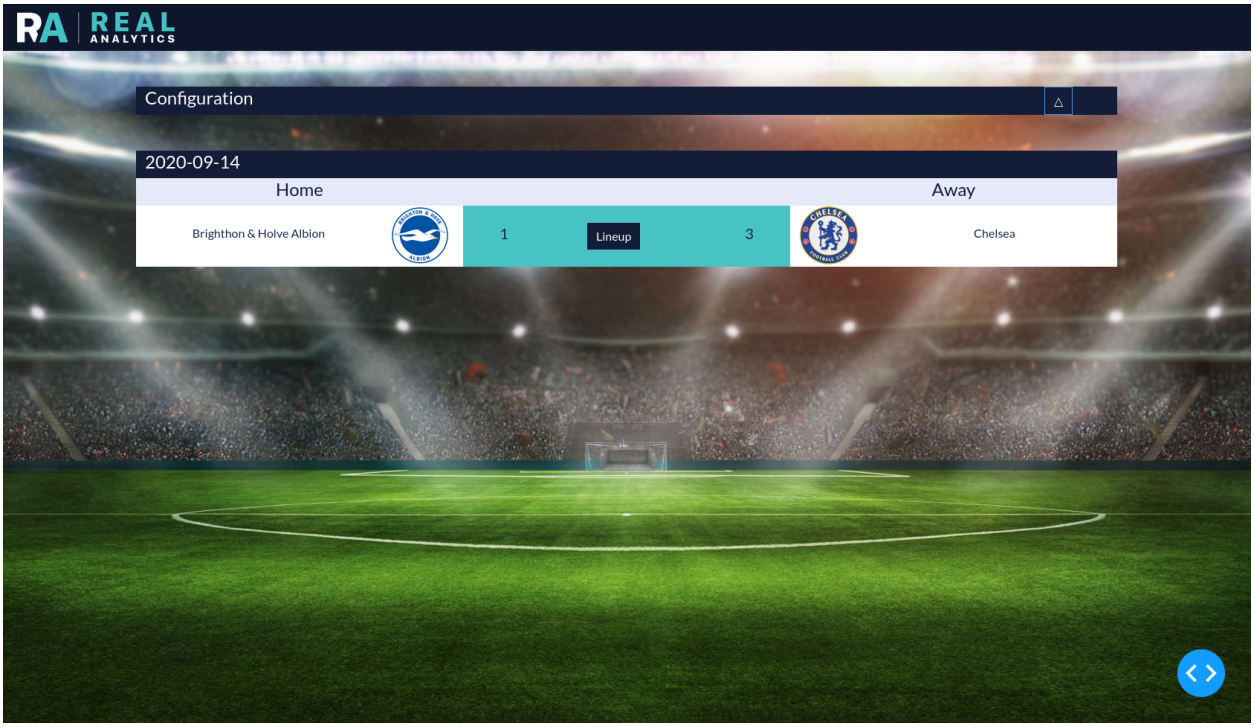

Fig. 4.1: The landing page.

4.3.2 Running configuration

It where the user can configurate each type of running.

4.3.3 Player page

It is the main page on which the player performance information will be displayed.

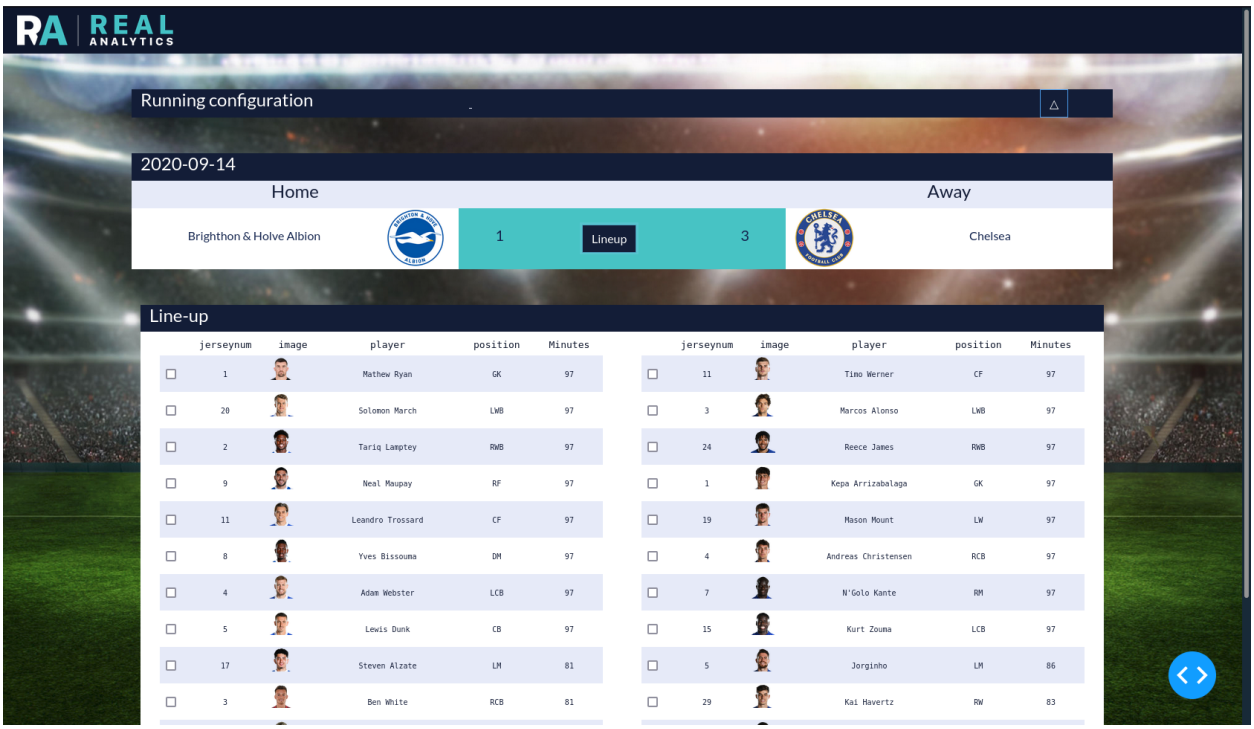

Fig. 4.2: Displayed content after clicking on line-up button.

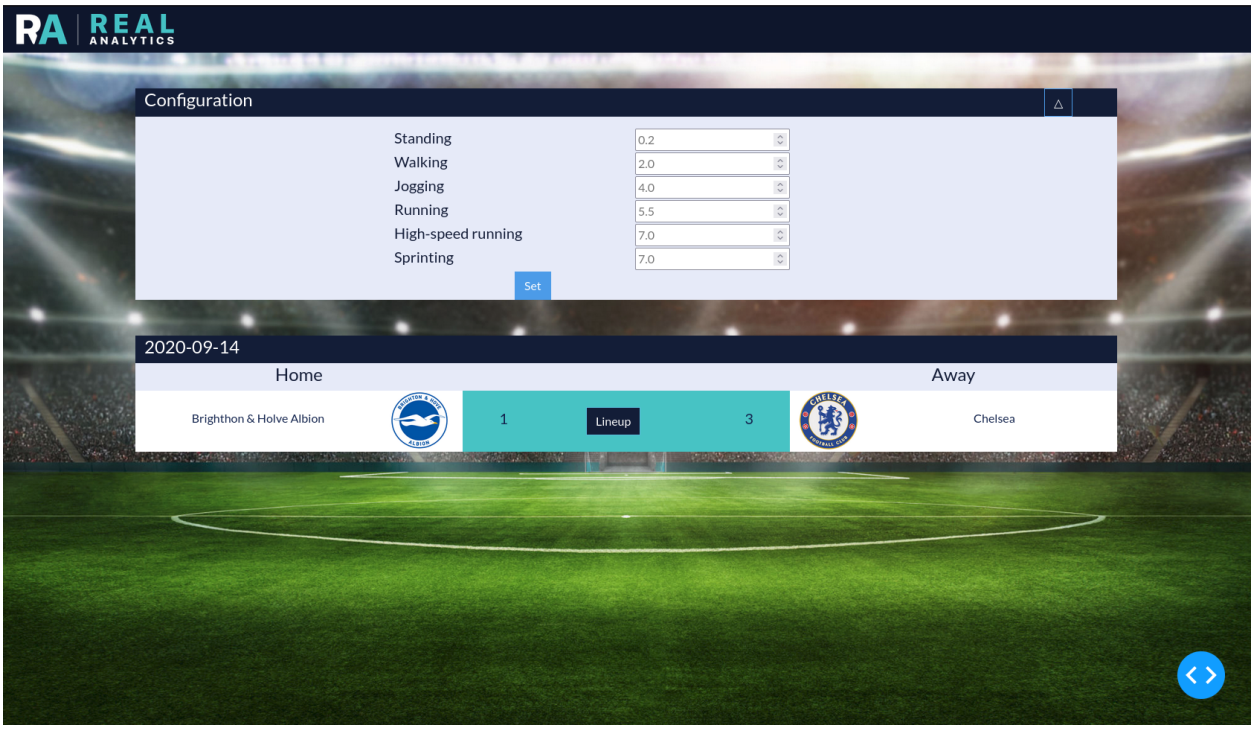

Fig. 4.3: Configuration of the runiing types.

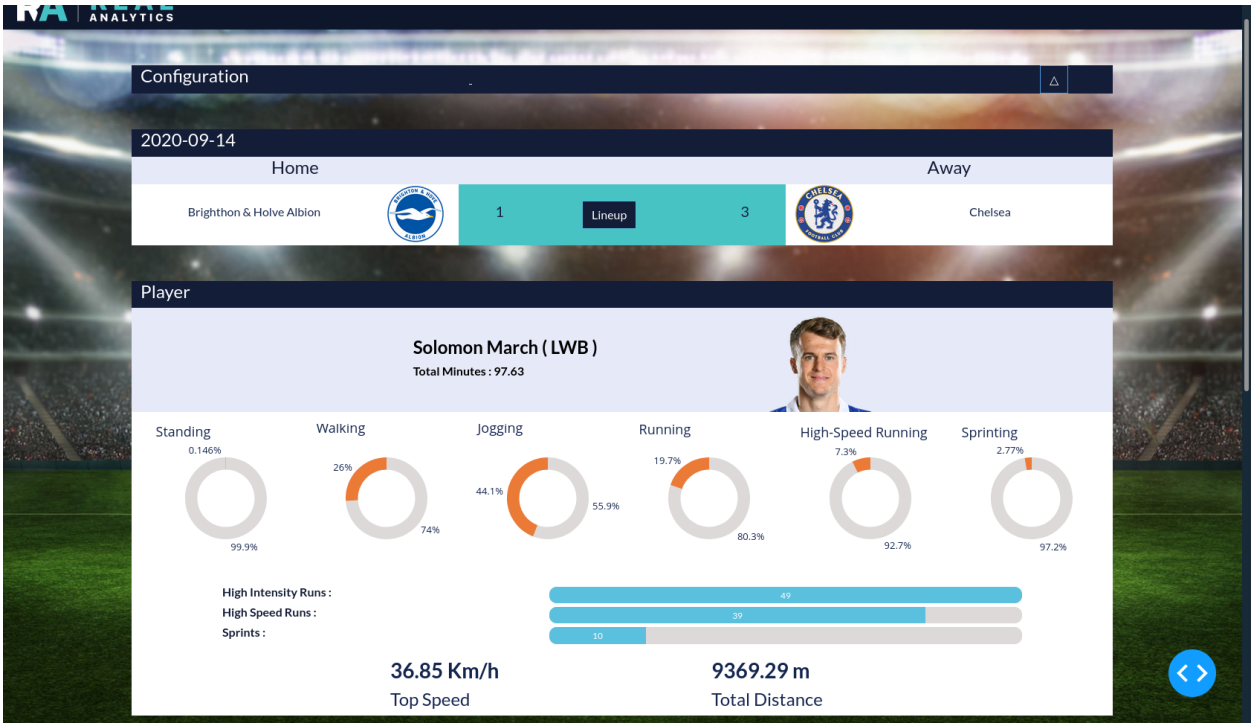

Fig. 4.4: Displayed content after selecting on one player .

4.3.4 Player Comparison

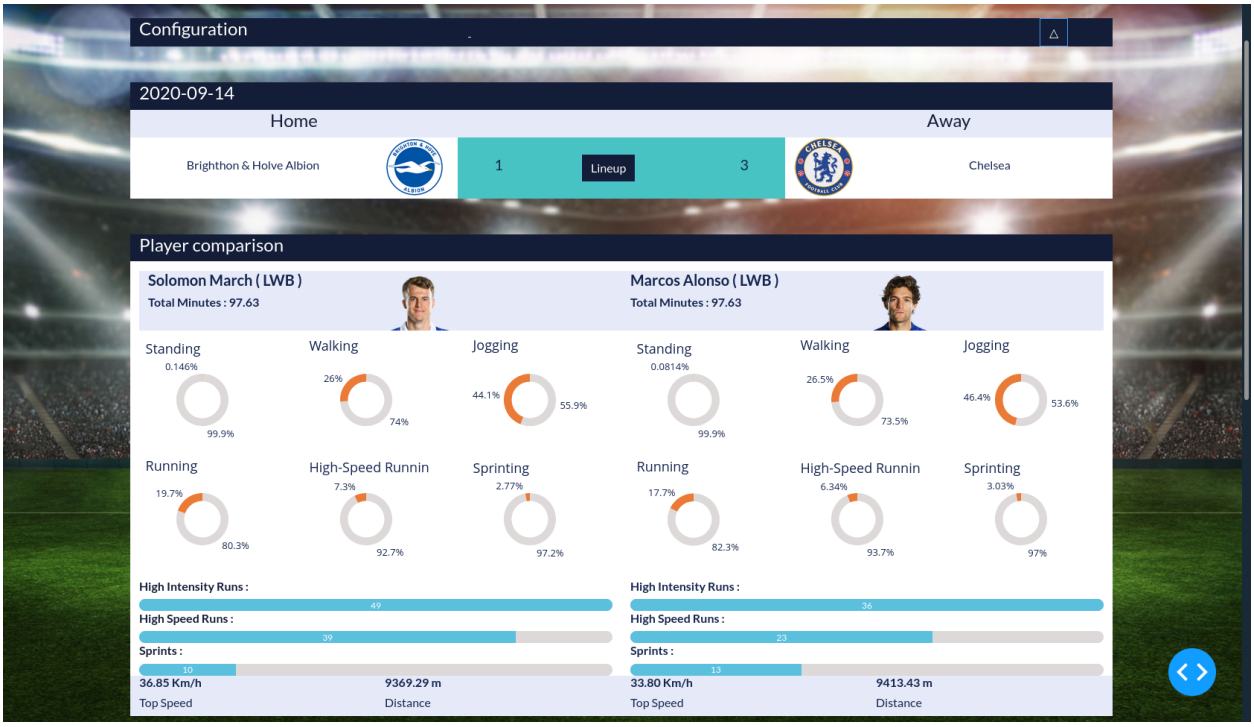

Fig. 4.5: Displayed content after selecting on two player.

CHAPTER

FIVE

CONCLUSION

Achieving this project has brought me many things, every part of the project challenged me to cope with new techniques and methods so that I could improve, even at times I faced struggles. By adapting the methdology scrum, the daily meetings helped me to strengthen my skills and to get motivated. This internship has introduced me to the working life and gave me a chance to put my theoretical knowledge to work.