

Football Analytics 101: Introduction to Football Analytics with Python

Ruslan Kornichuk | 17.06.2026

ver. 3.0.0



ANTONOV AN-124

Ruslan



● football analysis

Search term



● sports analytics

Search term



● football analyst

Search term



+ Add comparison

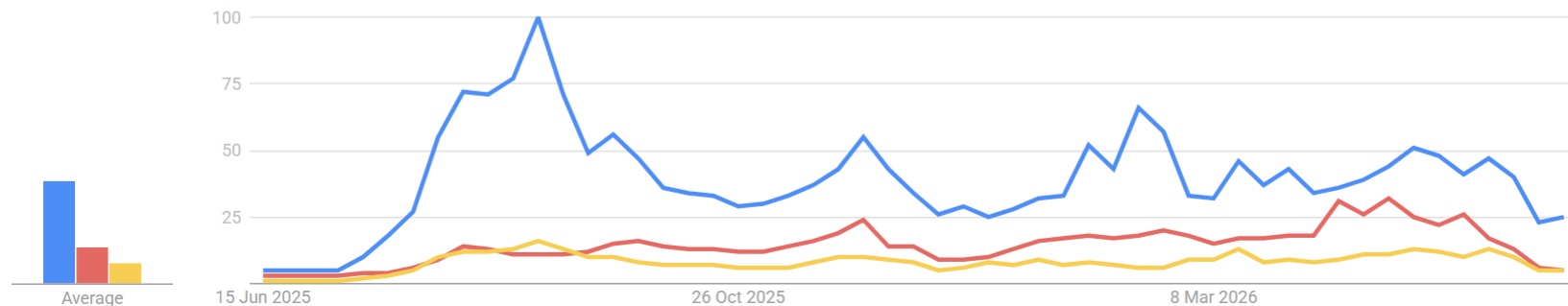
Worldwide ▼

Past 12 months ▼

All categories ▼

Web Search ▼

Interest over time ?



Average



Jagiellonia Białystok ✓

★ Favorites · June 8 at 10:53 PM · 🌐



Jagiellonia Białystok poszukuje osób na dwa stanowiska. Są to analityk danych pierwszego zespołu oraz analityk wideo pierwszego zespołu. Poniżej przedstawiamy szczegóły ofert. W przypadku obu ofert na zgłoszenia czekamy do 14 czerwca tego roku.

Analityk danych ➡ <https://bit.ly/OfertaPracyAnalitykDanych>

Analityk wideo ➡ <https://bit.ly/OfertaPracyAnalitykWideo>



52



1



8



TABLE OF CONTENTS

01

Hudl Statsbomb

02

Sportmonks

03

Understat

04

Extras

05

Learning materials

06

Closing slides

tinyurl.com/PYDATA26



01

Hudl Statsbomb

open-data

Public

Watch 229

Fork 935

Starred 3.3k

master

3 Branches

0 Tags

Go to file

Add file

<> Code

Statsbomb	Added 1647 new games, updated 1213 games	b0bc9f2 · 3 weeks ago	1,088 Commits
data	Added 1647 new games, updated 1213 games	3 weeks ago	
doc	Upload docs with new branding	4 years ago	
img	Delete statsbomb-logo.jpg	4 years ago	
.DS_Store	Added 370 new games, updated 1087 games.	3 years ago	
LICENSE.pdf	Update LICENSE.pdf	3 years ago	
README.md	Update README.md	4 years ago	

README

License

StatsBomb Open Data

Welcome to the StatsBomb Open Data repository.

StatsBomb are committed to sharing new data and research publicly to enhance understanding of the game of Football. We want to actively encourage new research and analysis at all levels. Therefore we have made certain leagues of StatsBomb Data freely available for public use for research projects and genuine interest in football analytics.

StatsBomb are hoping that by making data freely available, we will extend the wider football analytics community and

About

Free football data from StatsBomb

statsbomb.com/resource-centre/

- soccer
- open-data
- football-data
- football
- sports-stats
- sports-data

- Readme
- Activity
- Custom properties
- 3.3k stars
- 229 watching
- 935 forks
- Report repository

Releases

No releases published

Packages

No packages published

Contributors 5



Hudl Statsbomb Open Data structure

The data is provided as **JSON files** exported from the Statsbomb Data API, in the following structure:

- competition and seasons stored in **competitions.json**
- matches for each competition and **season**, stored in the **matches** folder. Each folder within is named for a competition ID, each file is named for a season ID within that competition
- events and lineups for each match, stored in **events** and **lineups** respectively. Each file is named for a match ID



statsbomb / open-data

Type / to search



Code

Issues 37

Pull requests 5

Agents

Security and quality

Insights



Files



master



Go to file

T

data

events

lineups

matches

three-sixty

competitions.json

doc

img

.DS_Store

LICENSE.pdf

README.md

open-data / data / competitions.json



Statsbomb Added 1647 new games, updated 1213 games

b0bc9f2 · 3 weeks ago

History

Code

Blame



Raw



```
1  [ {
2    "competition_id" : 9,
3    "season_id" : 281,
4    "country_name" : "Germany",
5    "competition_name" : "1. Bundesliga",
6    "competition_gender" : "male",
7    "competition_youth" : false,
8    "competition_international" : false,
9    "season_name" : "2023/2024",
10   "match_updated" : "2024-09-28T20:46:38.893391",
11   "match_updated_360" : "2025-11-15T23:17:41.827093",
12   "match_available_360" : "2025-11-15T23:17:41.827093",
13   "match_available" : "2024-09-28T20:46:38.893391"
14 }, {
15   "competition_id" : 9,
16   "season_id" : 27,
17   "country_name" : "Germany",
```



Hudl Statsbomb: Competitions

statsbomb_0001.ipynb

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

[2]: pd.set_option("display.max.columns", None)
pd.set_option("display.max.rows", None)
```

Load competitions.json file from Hudl Statsbomb GitHub and transform to Pandas DataFrame

```
[3]: url = "https://raw.githubusercontent.com/statsbomb/open-data/refs/heads/master/data/competitions.json"
data = requests.get(url).json()
competitions = pd.json_normalize(data)
competitions.head(3)
```

	competition_id	season_id	country_name	competition_name	competition_gender	competition_youth	competition_international	season_name	match_updated	match_updated
0	9	281	Germany	1. Bundesliga	male	False	False	2023/2024	2024-07-15T14:15:54.671676	2024-07-15T14:17:00.1
1	9	27	Germany	1. Bundesliga	male	False	False	2015/2016	2024-05-19T11:11:14.192381	
2	1267	107	Africa	African Cup of Nations	male	False	True	2023	2024-06-13T07:51:02.452825	

Competitions with season_name, competition_id, and season_id

```
[4]: competitions[["competition_name", "season_name", "competition_id", "season_id"]].head(3)
```

	competition_name	season_name	competition_id	season_id
0	1. Bundesliga	2023/2024	9	281
1	1. Bundesliga	2015/2016	9	27
2	African Cup of Nations	2023	1267	107

Hudl Statsbomb: Matches

statsbomb_0002.ipynb

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests
```

FIFA World Cup 2022

```
[2]: competition_id = 43
season_id = 106
```

```
[3]: url = f"https://raw.githubusercontent.com/statsbomb/open-data/refs/heads/master/data/matches/{competition_id}/{season_id}.json"
data = requests.get(url).json()
matches = pd.json_normalize(data)
matches.head(2)
```

	match_id	match_date	kick_off	home_score	away_score	match_status	match_status_360	last_updated	last_updated_360	match_week	...	competition_stage_id
0	3857256	2022-12-02	21:00:00.000	2	3	available	available	2023-02-17T23:45:15.306706	2023-04-26T23:49:58.956186	3	...	10
1	3869151	2022-12-03	21:00:00.000	2	1	available	available	2023-07-30T07:46:05.382784	2023-07-30T07:48:51.865595	4	...	33

2 rows x 42 columns

```
[4]: len(matches)
```

```
[4]: 64
```

```
[5]: for _, row in matches.head(2).iterrows():
    print(
        f"{row['home_team.home_team_name']} {row['away_team.away_team_name']} "
        f"{row['home_score']}:{row['away_score']} ({row['match_id']})"
    )
```

Serbia Switzerland 2:3 (3857256)
Argentina Australia 2:1 (3869151)

Hudl Statsbomb: Matches (filtered)

```
statsbomb_0003.ipynb x +
[1]: import pandas as pd
import requests

FIFA World Cup 2022

[2]: competition_id = 43
season_id = 106

[3]: url = f"https://raw.githubusercontent.com/statsbomb/open-data/refs/heads/master/data/matches/{competition_id}/{season_id}.json"
data = requests.get(url).json()
matches = pd.json_normalize(data)

Germany

[4]: team = "Germany"

filtered = matches[(matches["home_team.home_team_name"] == team) | (matches["away_team.away_team_name"] == team)]

for _, row in filtered.iterrows():
    print(
        f"{row['home_team.home_team_name']} {row['away_team.away_team_name']} "
        f"{row['home_score']}:{row['away_score']} ({row['match_id']})"
    )

Spain Germany 1:1 (3857263)
Germany Japan 1:2 (3857284)
Costa Rica Germany 2:4 (3857292)
```

Statsbomb event data

Hudl Statsbomb: Match events

statsbomb_0004.ipynb

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

Costa Rica 2:4 Germany

[2]: match_id = 3857292

[3]: url = f"https://raw.githubusercontent.com/statsbomb/open-data/refs/heads/master/data/events/{match_id}.json"
data = requests.get(url).json()
events = pd.json_normalize(data)

[4]: len(events)

[4]: 3869

[5]: events.head(3)
```

	id	index	period	timestamp	minute	second	possession	duration	type.id	type.name	...	substitution.outcome.name	substitution.replacement.id	substitution.replacement.name
0	2194e9dd-089b-44f8-a5c7-60436a00f0f9	1	1	00:00:00.000	0	0	1	0.0	35	Starting XI	...	NaN	NaN	
1	1247d52e-3c53-4106-bcf9-36c14a0c352d	2	1	00:00:00.000	0	0	1	0.0	35	Starting XI	...	NaN	NaN	
2	5ba53280-4efe-4482-b727-1328f6654d2a	3	1	00:00:00.000	0	0	1	0.0	18	Half Start	...	NaN	NaN	

3 rows × 116 columns

```
[6]: events.to_pickle("events.pkl")
```

Hudl Statsbomb: Event types

```
statsbomb_0005.ipynb x +
[1]: import pandas as pd

[2]: events = pd.read_pickle("events.pkl")

Event type frequency

[3]: events["type.name"].value_counts().head(5)

[3]: type.name
Pass      1140
Ball Receipt* 1062
Carry      868
Pressure   265
Ball Recovery 105
Name: count, dtype: int64

Event type frequency (Germany)

[4]: events_de = events.loc[events["team.name"] == "Germany"]
events_de["type.name"].value_counts().head(5)

[4]: type.name
Pass      766
Ball Receipt* 724
Carry      591
Pressure   140
Ball Recovery  57
Name: count, dtype: int64

[5]: events_de.to_pickle("events_de.pkl")
```

Hudl Statsbomb: Event types

```
statsbomb_0006.ipynb
```

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd
```

```
[2]: events_de = pd.read_pickle("events_de.pkl")
```

Passes (Germany)

```
[3]: passes_de = events_de.loc[events_de['type.name'] == 'Pass']
     passes_de[["player.name", "pass.recipient.name", "location", "pass.end_location"]].head(3)
```

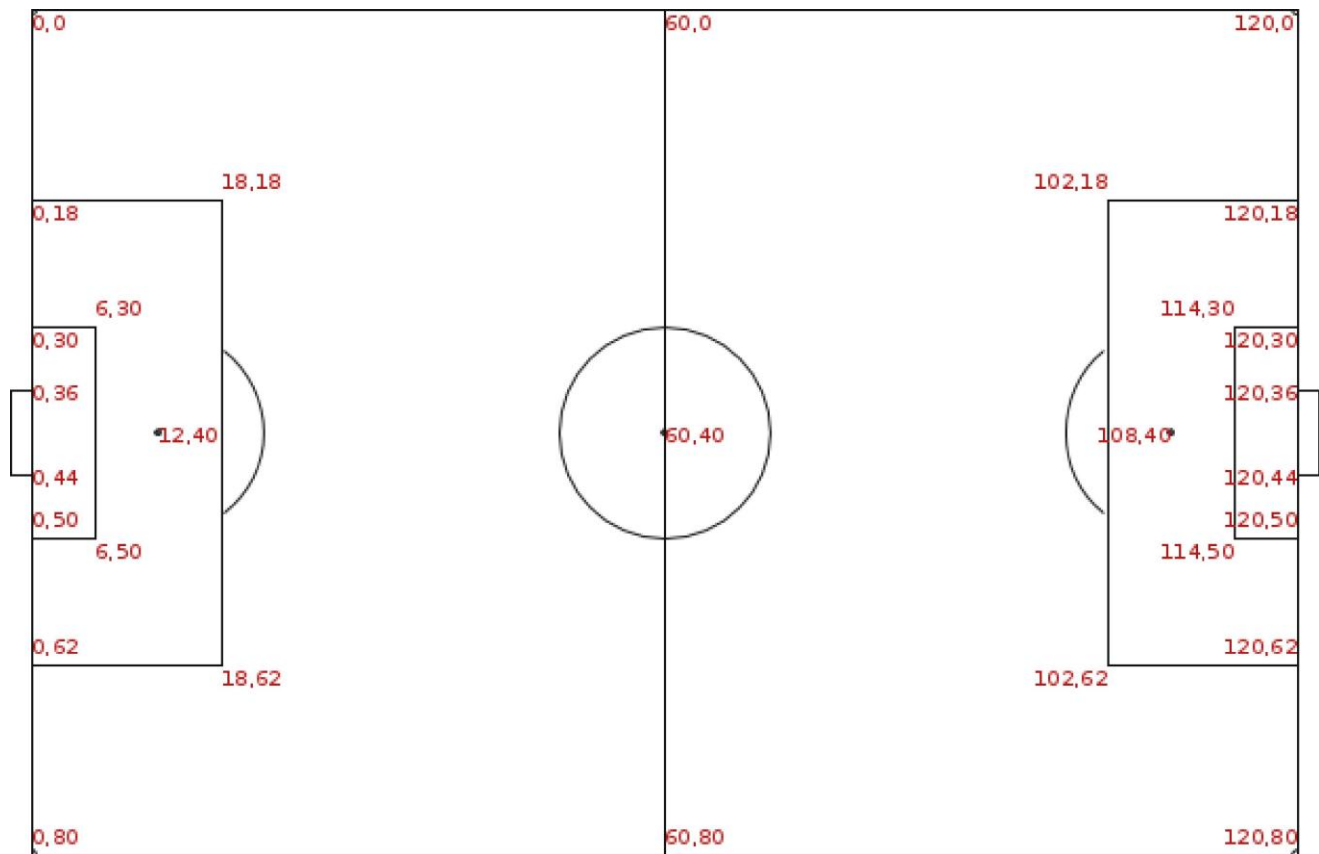
	player.name	pass.recipient.name	location	pass.end_location
4	Thomas Müller	Antonio Rüdiger	[61.0, 40.1]	[40.1, 36.2]
7	Antonio Rüdiger	Joshua Kimmich	[39.8, 36.4]	[40.6, 58.5]
10	Joshua Kimmich	Thomas Müller	[47.2, 62.2]	[89.9, 45.4]

Most frequent passing combinations between players

```
[4]: passes_de[["player.name", "pass.recipient.name"]].value_counts().head(10)
```

player.name	pass.recipient.name	
Joshua Kimmich	Niklas Süle	29
Niklas Süle	Joshua Kimmich	26
	Antonio Rüdiger	22
Antonio Rüdiger	Niklas Süle	20
Niklas Süle	İlkay Gündoğan	18
Antonio Rüdiger	David Raum	16
İlkay Gündoğan	Niklas Süle	15
David Raum	Antonio Rüdiger	15
Joshua Kimmich	Lukas Klostermann	14
Antonio Rüdiger	Joshua Kimmich	14

Name: count, dtype: int64



Hudl Statsbomb: Team's formation

```
statsbomb_0007.ipynb x +
[1]: import pandas as pd

[2]: events_de = pd.read_pickle("events_de.pkl")

Team's formation (Germany)

[3]: events_de['type.id'].isin([35]).any() # starting eleven

[3]: True

[4]: events_de['type.id'].isin([36]).any() # tactical shift

[4]: True

[5]: events_de[events_de['type.id'] == 35]["tactics.formation"].astype(int).astype(str)

[5]: 1      4231
      Name: tactics.formation, dtype: object

[6]: tactical_shifts = events_de[events_de['type.id'] == 36]
      tactical_shifts = tactical_shifts.copy()
      tactical_shifts["tactics.formation"] = tactical_shifts["tactics.formation"].astype(int).astype(str)

[7]: tactical_shifts[["type.name", "tactics.formation"]]

[7]:
   type.name  tactics.formation
1944 Tactical Shift             4231
2384 Tactical Shift             442
2757 Tactical Shift             352

[8]: tactical_shifts.to_pickle("tactical_shifts.pkl")
```

Hudl Statsbomb: Team's formation

statsbomb_0008.ipynb

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd

[2]: tactical_shifts = pd.read_pickle("tactical_shifts.pkl")

[3]: exploded = tactical_shifts.explode('tactics.lineup').reset_index(drop=True)
normalized = pd.json_normalize(exploded["tactics.lineup"])
pd.concat([exploded[["type.name", "tactics.formation"]], normalized], axis=1).head(11)
```

	type.name	tactics.formation	jersey_number	player.id	player.name	position.id	position.name
0	Tactical Shift	4231	1	5570	Manuel Neuer	1	Goalkeeper
1	Tactical Shift	4231	16	8507	Lukas Klostermann	2	Right Back
2	Tactical Shift	4231	15	6322	Niklas Süle	3	Right Center Back
3	Tactical Shift	4231	2	3167	Antonio Rüdiger	5	Left Center Back
4	Tactical Shift	4231	3	12034	David Raum	6	Left Back
5	Tactical Shift	4231	6	5579	Joshua Kimmich	9	Right Defensive Midfield
6	Tactical Shift	4231	21	10287	İlkay Gündoğan	11	Left Defensive Midfield
7	Tactical Shift	4231	19	3053	Leroy Sané	17	Right Wing
8	Tactical Shift	4231	14	39565	Jamal Musiala	19	Center Attacking Midfield
9	Tactical Shift	4231	10	8400	Serge Gnabry	21	Left Wing
10	Tactical Shift	4231	13	5562	Thomas Müller	23	Center Forward

position number	position abbreviation	position name
1	GK	Goalkeeper
2	RB	Right Back
3	RCB	Right Center Back
4	CB	Center Back
5	LCB	Left Center Back
6	LB	Left Back
7	RWB	Right Wing Back
8	LWB	Left Wing Back
9	RDM	Right Defensive Midfield
10	CDM	Center Defensive Midfield
11	LDM	Left Defensive Midfield
12	RM	Right Midfield
13	RCM	Right Center Midfield
14	CM	Center Midfield
15	LCM	Left Center Midfield

position number	position abbreviation	position name
16	LM	Left Midfield
17	RW	Right Wing
18	RAM	Right Attacking Midfield
19	CAM	Center Attacking Midfield
20	LAM	Left Attacking Midfield
21	LW	Left Wing
22	RCF	Right Center Forward
23	ST	Striker
24	LCF	Left Center Forward
25	SS	Secondary Striker

Hudl IQ

(American football)



The unrivaled standard for verified football data.

Powered by AI, computer vision and expert data collectors, Hudl IQ delivers **10x more data** than any one provider, and in less time.



Save Hours Every Week



Advanced Physical Metrics



Revolutionary Recruiting Tools



Dedicated Analysis Platforms



Commitment to Your Privacy



**Transparent Data
Collection Methods**

Get the most out of Hudl IQ.

02

Sportmonks



FREE SUBSCRIPTION

Football

Cricket



Football Free Plan

4 leagues 

API Calls

3000 API Calls

Ready to explore more leagues and more data features?

[Start new Subscription](#)

Manage

[Manage Team](#)[Get support](#)



Football



4 league



API Calls

3000 API

Ready to explore more

MY LEAGUES



Football Free Plan

Search type to search all leagues and countries...

Total leagues in plan: 4

#ID

 **Denmark #320**

Superliga

271

Superliga Play-offs

1659

 **Scotland #1161**

Premiership

501

Premiership Play-Offs

513

[Visit our Coverage page](#)

Close

Manage

[Manage Team](#)



[Get support](#)



API STRUCTURE

Subscription and API

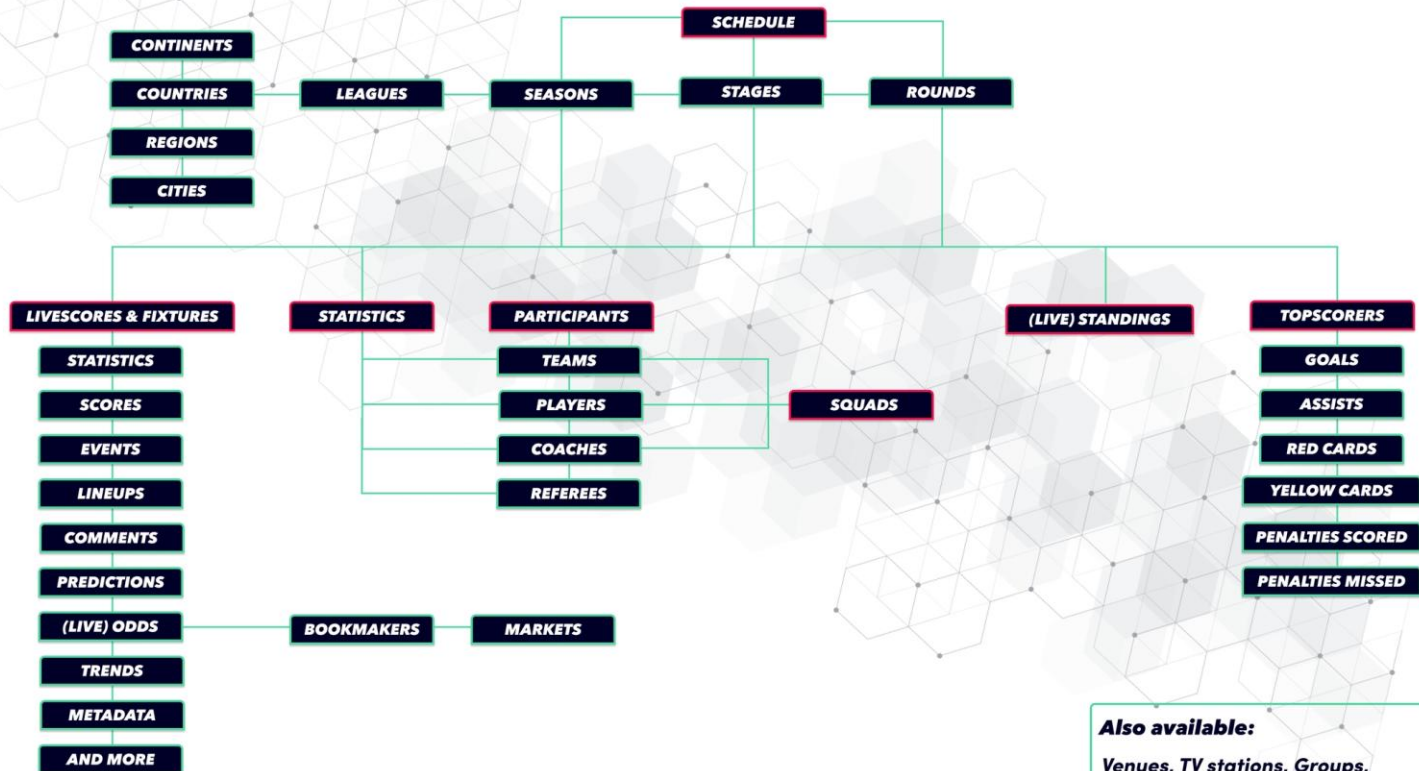
ENRICHMENTS

FILTERS

RESOURCES

TYPES

LEAGUES



Also available:

Venues, TV stations, Groups,
Transfers, Rivals and More

Sportmonks: Leagues and countries

```
sportmonks_0001.ipynb
```

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX208tyVADTRuvyBaGmLVrZkwcwrhfbRjR9T5ZAQWXovTUEBytl4onZ"
params = {"api_token": API_TOKEN}

Get leagues

[3]: leagues = requests.get("https://api.sportmonks.com/v3/football/leagues", params=params).json()["data"]
len(leagues)

[3]: 4

[4]: country_ids = set()
for league in leagues:
    country_id = league["country_id"]
    country_ids.add(country_id)

[5]: country_ids

[5]: {320, 1161}

Get countries

[6]: for country_id in country_ids:
    name = requests.get(f"https://api.sportmonks.com/v3/core/countries/{country_id}", params=params).json()["data"]["name"]
    print(name)

Denmark
Scotland
```

Sportmonks: Fixtures (matches) by date

sportmonks_0002.ipynb

Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX200tyVADTRuvyBaGmLvRZkwcwrhfBjR9T5ZAQWXovTUEBytL4onZ"
params = {"api_token": API_TOKEN}

Fixtures (matches) by date ( YYYY-MM-DD )

[3]: date = "2025-04-21"
data= requests.get(f"https://api.sportmonks.com/v3/football/fixtures/date/{date}", params=params).json()["data"]
matches = pd.json_normalize(data)
matches
```

	id	sport_id	league_id	season_id	stage_id	group_id	aggregate_id	round_id	state_id	venue_id	name	starting_at	result_info	leg	details	length	placeholder	has_odds	has
0	19404981	1	271	23584	77471147	None	None	369912	5	4832	Vejle Boldklub vs Silkeborg IF	2025-04-21 12:00:00	Silkeborg IF won after full-time.	1/1	None	90	False	True	
1	19404982	1	271	23584	77471147	None	None	369912	5	1467	Viborg FF vs Lyngby Boldklub	2025-04-21 12:00:00	Game ended in draw.	1/1	None	90	False	True	
2	19404953	1	271	23584	77471146	None	None	369902	5	5659	Brøndby IF vs Randers FC	2025-04-21 14:00:00	Brøndby IF won after full-time.	1/1	None	90	False	True	
3	19404954	1	271	23584	77471146	None	None	369902	5	5655	FC København vs AGF	2025-04-21 16:00:00	FC København won after full-time.	1/1	None	90	False	True	

Sportmonks: Fixtures (matches) by date range

sportmonks_0003.ipynb

Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "SipsUl8eBfEIm0uqoUxQZwXRchC3UNJ88qHt6HUcFfU5Lj6hWagNancfuVBE"

Fixtures (matches) by date range (max 100 days)

[3]: start_date = "2025-01-01"
end_date = "2025-04-01"

data = []
page = 1
has_more = True

while has_more:
    params = {"api_token": API_TOKEN, "per_page": 50, "page": page}
    r = requests.get(f"https://api.sportmonks.com/v3/football/fixtures/between/{start_date}/{end_date}", params=params).json()
    data.extend(r["data"])
    has_more = r.get("pagination", {}).get("has_more", False)
    page += 1

[4]: matches = pd.json_normalize(data)
matches.head(2)
```

	id	sport_id	league_id	season_id	stage_id	group_id	aggregate_id	round_id	state_id	venue_id	name	starting_at	result_info	leg	details	length	placeholder	has_odds	h
0	19146817	1	501	23690	77471570	None	None	340593	5	8928	Aberdeen vs Ross County	2025-01-02 15:00:00	Ross County won after full-time.	1/1	None	90	False	True	
1	19146818	1	501	23690	77471570	None	None	340593	5	336296	Hearts vs Motherwell	2025-01-02 15:00:00	Hearts won after full-time.	1/1	None	90	False	True	

Sportmonks: Fixture (match) by ID

```
sportmonks_0004.ipynb
```

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX200tyVADTRuvyBaGmLVrZkwcwrhfbRjR9T5ZAQWXovTUEBytl4onZ"
params = {"api_token": API_TOKEN}

Aberdeen vs Ross County ( 19146817 )

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]

[4]: match = pd.json_normalize(data)

[5]: match.columns

[5]: Index(['id', 'sport_id', 'league_id', 'season_id', 'stage_id', 'group_id',
        'aggregate_id', 'round_id', 'state_id', 'venue_id', 'name',
        'starting_at', 'result_info', 'leg', 'details', 'length', 'placeholder',
        'has_odds', 'has_premium_odds', 'starting_at_timestamp'],
        dtype='object')
```

Sportmonks: Fixture (match) statistics

sportmonks_0005.ipynb

+

Python 3 (ipykernel)

Code

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX200tyVADTRuvyBaGmLVrZkwcwrhfbRjR9T5ZAQWXovTUEBytl4onZ"
params = {"api_token": API_TOKEN, "include": "statistics.type;events;formations;timeline;lineups.details.type"}

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]
match = pd.json_normalize(data)
match.columns

[3]: Index(['id', 'sport_id', 'league_id', 'season_id', 'stage_id', 'group_id',
        'aggregate_id', 'round_id', 'state_id', 'venue_id', 'name',
        'starting_at', 'result_info', 'leg', 'details', 'length', 'placeholder',
        'has_odds', 'has_premium_odds', 'starting_at_timestamp', 'statistics',
        'events', 'formations', 'timeline', 'lineups'],
        dtype='object')

[4]: statistics = pd.json_normalize(match.statistics[0])

[5]: statistics.head(5)
```

	id	fixture_id	type_id	participant_id	location	data.value	type.id	type.name	type.code	type.developer_name	type.model_type	type.stat_group
0	207022998	19146817	34	273	home	9	34	Corners	corners	CORNERS	statistic	offensive
1	207023003	19146817	84	273	home	2	84	Yellowcards	yellowcards	YELLOWCARDS	statistic	overall
2	207023280	19146817	1605	273	home	33	1605	Successful Dribbles Percentage	successful-dribbles-percentage	SUCCESSFUL_DRIBBLES_PERCENTAGE	statistic	offensive
3	207023287	19146817	1605	246	away	38	1605	Successful Dribbles Percentage	successful-dribbles-percentage	SUCCESSFUL_DRIBBLES_PERCENTAGE	statistic	offensive
4	207023271	19146817	45	273	home	66	45	Ball Possession %	ball-possession	BALL_POSSESSION	statistic	overall

sport round stage group aggregate league season venue state
weatherReport lineups events timeline comments trends statistics
periods participants odds premiumOdds ↗ inplayOdds postmatchNews
prematchNews metadata tvStations predictions referees formations
ballCoordinates sidelined metadata scores

Sportmonks: Fixture (match) events

sportmonks_0006.ipynb

+

Python 3 (ipykernel)

Code

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX200tyVADTRuvyBaGmLVrZkwcwrhfbRjR9T5ZAQWXovTUEBytl4onZ"
params = {"api_token": API_TOKEN, "include": "events"}

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]
match = pd.json_normalize(data)

[4]: events = pd.json_normalize(match.events[0])
len(events)

[4]: 17

[5]: events.head(5)
```

	id	fixture_id	period_id	participant_id	type_id	section	player_id	related_player_id	player_name	related_player_name	...	info	addition	minute	extra_minute	injured	on_ber
0	148273586	19146817	5800865	273	18	event	173059	464190.0	Kevin Nisbet	Ester Sokler	...	None	1st Substitution	30	NaN	None	Fa
1	148273591	19146817	5800865	246	19	event	123690	NaN	James Brown	None	...	Foul	2nd Yellowcard	31	NaN	None	Fa
2	148273534	19146817	5800865	273	19	event	37344344	NaN	T. Keskinen	None	...	Foul	1st Yellowcard	23	NaN	None	Fa
3	148273728	19146817	5800865	273	14	event	173059	NaN	Kevin Nisbet	None	...	Field Goal	2nd Goal	45	11.0	None	Fa
4	148273645	19146817	5800865	246	18	event	4893665	546803.0	George Harmon	Michee Efete	...	None	2nd Substitution	44	NaN	None	Fa

5 rows × 21 columns

Sportmonks: Fixture (match) timeline

sportmonks_0008.ipynb

Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "PTdd20P9btErrQPcRSTXpxLtfL2iIwz70Wi8LrLg9jhw51Mt0C9CtTuWzP6Y"
params = {"api_token": API_TOKEN, "include": "timeline"}

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]
match = pd.json_normalize(data)

[4]: timeline = pd.json_normalize(match.timeline[0])
len(timeline)

[4]: 17

[5]: def head_skip_all_nan(df, n):
    return df.dropna(axis=1, how='all').head(n)

head_skip_all_nan(timeline, 5)

[5]:
```

	id	fixture_id	period_id	participant_id	type_id	section	addition	minute	extra_minute	on_bench	sort_order
0	148273597	19146817	5800865	246	126	timeline	3rd Corner	37	NaN	False	3
1	148273435	19146817	5800865	246	126	timeline	1st Corner	6	NaN	False	1
2	148273535	19146817	5800865	246	126	timeline	2nd Corner	24	NaN	False	2
3	148273629	19146817	5800865	273	126	timeline	6th Corner	38	NaN	False	6
4	148273735	19146817	5800865	273	126	timeline	8th Corner	45	NaN	False	8

Sportmonks: Formations

```
sportmonks_0007.ipynb
```

Open in... Python 3 (ipykernel)

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "tqz8BHX200tyVADTRuvyBaGmLVrZkwcwrhfbRjR9T5ZAQWXovTUEBytl4onZ"
params = {"api_token": API_TOKEN, "include": "formations"}

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]
match = pd.json_normalize(data)

[4]: formations = pd.json_normalize(match.formations[0])

[5]: formations.head(5)
```

	id	fixture_id	participant_id	formation	location
0	8387600	19146817	246	3-5-1-1	away
1	8387601	19146817	273	4-2-3-1	home

Sportmonks: Fixture (match) lineups

```
sportmonks_0009.ipynb
```

```
[1]: import pandas as pd
import requests

[2]: API_TOKEN = "PTdd20P9btERrQPcRSTXpxLtlfL2iIwz70Wi8LrLg9jhw51Mt0C9CtTuWzP6Y"
params = {"api_token": API_TOKEN, "include": "lineups.details.type"}

[3]: data = requests.get("https://api.sportmonks.com/v3/football/fixtures/19146817", params=params).json()["data"]
match = pd.json_normalize(data)

[4]: lineups = pd.json_normalize(match.lineups[0])
len(lineups)

[4]: 40

[5]: def head_skip_all_nan(df, n):
    return df.dropna(axis=1, how='all').head(n)

head_skip_all_nan(lineups.drop('details', axis=1), 5)

[5]:
```

	id	sport_id	fixture_id	player_id	team_id	position_id	type_id	formation_position	player_name	jersey_number
0	14597927439	1	19146817	123690	246	26	11	5.0	James Brown	2
1	14597927412	1	19146817	580993	246	25	11	4.0	Ryan David Leak	3
2	14597927435	1	19146817	9654	246	25	11	2.0	Akil Wright	4
3	14597927126	1	19146817	174741	246	26	12	NaN	Scott Allardice	6
4	14597927130	1	19146817	34422813	246	26	12	NaN	Victor Loturi	7

```
[6]: lineups.to_pickle("lineups.pkl")
```

Sportmonks: Fixture (match) lineups details

sportmonks_0010.ipynb

+

Python 3 (ipykernel)

Code

```
[1]: import pandas as pd

[2]: lineups = pd.read_pickle("lineups.pkl")

[3]: exploded = lineups.explode('details').reset_index(drop=True)
      normalized = pd.json_normalize(exploded["details"])
      pd.concat([exploded.drop('details', axis=1), normalized[["data.value", "type.name"]]], axis=1)
```

	id	sport_id	fixture_id	player_id	team_id	position_id	formation_field	type_id	formation_position	player_name	jersey_number	data.value	type.name
0	14597927439	1	19146817	123690	246	26.0	3:5	11	5.0	James Brown	2	75	Accurate Passes Percentage
1	14597927439	1	19146817	123690	246	26.0	3:5	11	5.0	James Brown	2	4	Duels Won
2	14597927439	1	19146817	123690	246	26.0	3:5	11	5.0	James Brown	2	2	Fouls Drawn
3	14597927439	1	19146817	123690	246	26.0	3:5	11	5.0	James Brown	2	6	Total Duels
4	14597927439	1	19146817	123690	246	26.0	3:5	11	5.0	James Brown	2	12	Accurate Passes
...	...	...	...	...	...	...	...	...	...	...	...	...	...
922	14597927129	1	19146817	37344344	273	27.0	4:3	11	10.0	Topi Keskinen	81	5	Turn Over
923	14597927129	1	19146817	37344344	273	27.0	4:3	11	10.0	Topi Keskinen	81	22	Passes
924	14597927129	1	19146817	37344344	273	27.0	4:3	11	10.0	Topi Keskinen	81	1	Aerials Won
925	14597927129	1	19146817	37344344	273	27.0	4:3	11	10.0	Topi Keskinen	81	3	Chances Created
926	14597927129	1	19146817	37344344	273	27.0	4:3	11	10.0	Topi Keskinen	81	110	Cumulative Minutes Played

927 rows × 13 columns

03

Understat

Expected goals (xG) is the new revolutionary football metric, which allows you to evaluate team and player performance.

In a low-scoring game such as football, final match score does not provide a clear picture of performance.

This is why more and more sports analytics turn to the advanced models like xG, which is a statistical measure of the quality of chances created and conceded.

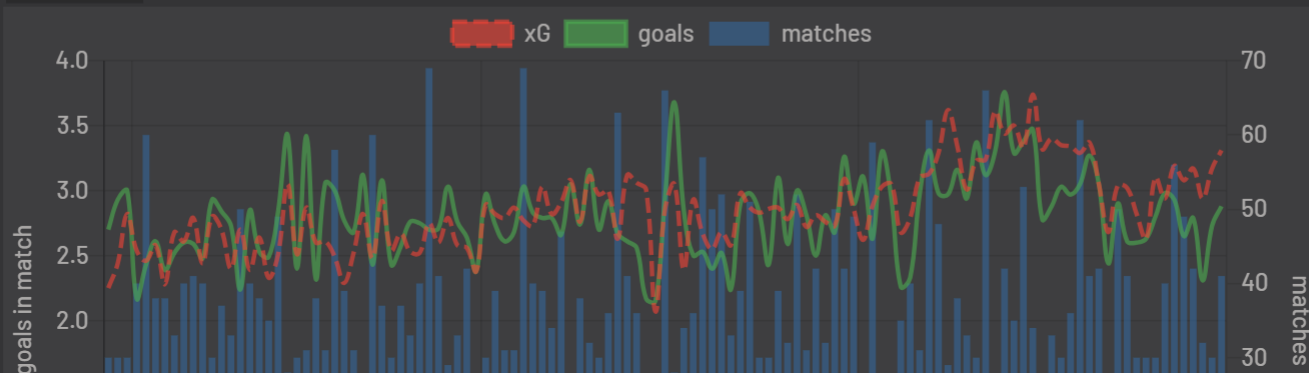
Our goal was to create the most precise method for shot quality evaluation.

For this case, we trained neural network prediction algorithms with the large dataset (>100,000 shots, over 10 parameters for each).

On this site, you will find our detailed xG statistics for the top European leagues.

League

EPL ▼



04

Extras

 mplsoccer

Public

 Watch 16

 Fork 88

 Starred 520

 main

 16 Branches

 0 Tags

Q Go to file

T

Add file

<> Code

 andrewRowlinson	update python version to a minimum of 3.10 (#117)	64c1db1 · 8 months ago	🕒 695 Commits
docs	refactored mplsoccer for multiple sports (#115)	8 months ago	
examples	refactored mplsoccer for multiple sports (#115)	8 months ago	
mplsoccer	update python version to a minimum of 3.10 (#117)	8 months ago	
tests	refactored mplsoccer for multiple sports (#115)	8 months ago	
.gitignore	refactored mplsoccer for multiple sports (#115)	8 months ago	
.pylintrc	update .pylintrc and linting fixes.	4 years ago	
.readthedocs.yml	Install via uv rather than Conda. Fix docs.	last year	
CHANGELOG.md	update python version to a minimum of 3.10 (#117)	8 months ago	
LICENSE	Install via uv rather than Conda. Fix docs.	last year	
README.md	fix README version 1.2.1.	3 years ago	
pyproject.toml	update python version to a minimum of 3.10 (#117)	8 months ago	

README

MIT license



mplsoccer is a Python library for plotting soccer/football charts in Matplotlib and loading StatsBomb open-data.

Installation

About

Football pitch plotting library for matplotlib

- 📖 Readme
- 📄 MIT license
- 📈 Activity
- ☆ 520 stars
- 👁 16 watching
- 🍴 88 forks
- Report repository

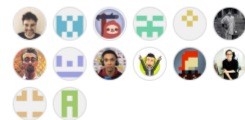
Releases

No releases published

Packages

No packages published

Contributors 19



+ 5 contributors

Languages

Python 100.0%



Enter Person, Team, Section, etc

Search

Players

Clubs

Competitions

Matches5

Awards

Countries

Stathead

Mailing List

Full Site Menu Below ▾

Football Stats and History Statistics, scores and history for 100+ men's and women's club and national team competitions.

2026 FIFA World Cup



2026 World Cup Tables and Stats

Team & Player Stats

[Standard Stats](#) · [Goalkeeping](#) · [Shooting](#) · [Playing Time](#) · [Miscellaneous Stats](#)

✉ [Sign up for our World Cup Newsletter!](#) New issues every [Monday, Wednesday, and Friday](#).

Group stage Begins: Thursday June 11, 2026

Knockout stage Begins: Saturday July 4, 2026

Third-place match: Saturday July 18, 2026

Final: Sunday July 19, 2026

[See All Fixtures](#)

[World Cup Players by League](#)

World Cup History on FBref

All World Cup Tournaments: [Men](#), [Women](#)

Previous Tournaments: [2022](#), [2018](#), [2014](#), [2010](#)

Previous Champions: [Argentina](#), [France](#), [Germany](#), [Spain](#)

All-Time Top Scorers: [Miroslav Klose](#), [Ronaldo](#), [Gerd Müller](#)

Most recent Hat Trick: [Kylian Mbappé](#) ([France](#), [December 18, 2022](#))

Leaders

Goals



[Folarin Balogun](#) [United States](#)

2



[Elijah Just](#) [New Zealand](#)

2

Assists



[Chris Wood](#) [New Zealand](#)

2



[Deniz Undav](#) [Germany](#)

2

Recent Results

Sunday, June 14

Germany	7–1	Curaçao
Netherlands	2–2	Japan
Côte d'Ivoire	1–0	Ecuador
Sweden	5–1	Tunisia

Monday, June 15

Spain	0–0	Cape Verde
Belgium	1–1	Egypt
IR Iran	2–2	New Zealand
Saudi Arabia	1–1	Uruguay

Upcoming Matches

Tuesday, June 16

France	15:00	(21:00)	Senegal
Iraq	18:00	(00:00)	Norway
Argentina	20:00	(03:00)	Algeria
Austria	21:00	(06:00)	Jordan

Wednesday, June 17

Portugal	12:00	(19:00)	Congo DR
England	15:00	(22:00)	Croatia
Ghana	19:00	(01:00)	Panama
Uzbekistan	20:00	(04:00)	Colombia

The dev-friendly football API

RESTful. Reliable. Free to use. Easy to integrate.

Did you realize the delay in the table below? That was the time it took to make a

```
GET https://api.football-data.org/v4/matches
```

in the background. But in the end it's that easy... one request.

Date/Status		Opponents		Score
FINISHED		 Argentina	vs.	 Algeria 3:0
FINISHED		 Austria	vs.	 Jordan 3:1
17/06/2026	17h00 TIMED	 Portugal	vs.	 Congo DR --
17/06/2026	20h00 TIMED	 England	vs.	 Croatia --
17/06/2026	23h00 TIMED	 Ghana	vs.	 Panama --

05

Learning materials

**IMDb**

All ▾



Sign in

Use app

< Ted Lasso

S3.E7 < All episodes >

All topics



The Strings That Bind Us

Episode aired Apr 26, 2023 · TV-MA · 57m



Comedy

Drama

Sport

The Greyhounds try a new strategy that has everyone thinking outside the box. Sam prepares to host a VIP guest at Ola's.

★ 8.3/10 8.4K

☆ Rate

Director [Matt Lipsey](#)**Writers** [Phoebe Walsh](#) · [Sasha Garron](#) · [Keeley Hazell](#) >**Stars** [Jason Sudeikis](#) · [Hannah Waddingham](#) · [Jeremy Swift](#) >

The Visionary: Rinus Michels

Who: The coach of Ajax and the Netherlands national team.

When: Late 1960s – 1970s.

How: Michels formalized "**Totaalvoetbal**" (Total Football). The core rule: any outfield player can take over the role of any other player in the team.

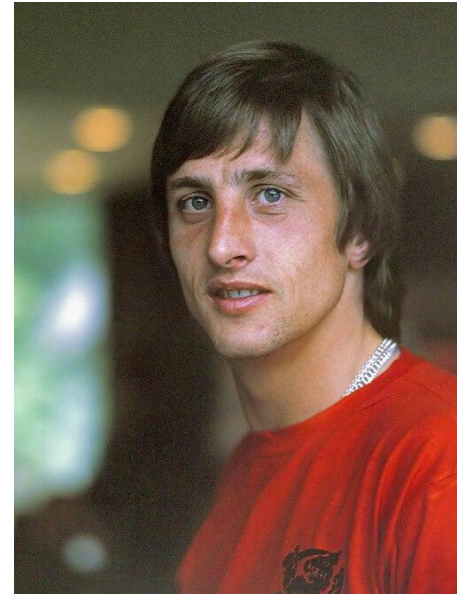


The Perfectionist: Johan Cruyff

Who: The star player of Ajax. The coach of Ajax and Barcelona.

When: 1970s (player) - 1990s (coach).

How: Cruyff prioritized space and technique, laying the foundation for Barcelona's "**tiki-taka**".



How To Get Started In Football Analytics

29 AUG 2024 7 MIN READ

Since our inception, Hudl Statsbomb has been at the forefront of educating the next generation of football analysts. We're committed to providing materials and resources to develop the skills needed to enter the industry, from free datasets and code to industry-standard education and training courses. We've opened the doors to our extensive knowledge base, all to help aspiring analysts hone their craft.

One of the most common questions we receive is, "How do I get started in football analytics?" This article is our answer — a comprehensive guide that gathers all our resources in one place, designed to be your go-to reference for both inspiration and education. Whether you're just starting out or looking to upskill, you'll find everything you need here.

How to Learn Sports Analytics for Beginners: A Step-by-Step Guide

By McKay Johns • November 24, 2024

If you feel lost with learning sports analytics, let's dive into the step by step guide on how to do it.

[Python](#)[R](#)[sports analytics](#)[coding](#)

Sports analytics has revolutionized how teams, coaches, and fans understand the game. From predicting player performance to optimizing game strategies, analytics plays a pivotal role in modern sports. If you're a beginner intrigued by the idea of analyzing game data and uncovering actionable insights, this guide will show you how to get started.

What is Sports Analytics?



About

Outcomes

Courses

Testimonials

Specialization - 5 course series

Sports analytics has emerged as a field of research with increasing popularity propelled, in part, by the real-world success illustrated by the best-selling book and motion picture, Moneyball. Analysis of team and player performance data has continued to revolutionize the sports industry on the field,

[Read more](#)



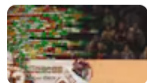
Foundations of Sports Analytics: Data, Represe...

Course 1 • 49 hours



Moneyball and Beyond

Course 2 • 28 hours



Prediction Models with Sports Data

Course 3 • 33 hours



Wearable Technologies and Sports Analytics

Instructors



Stefan Szymanski

University of Michigan

3 Courses • 29,998 learners



Youngho Park

University of Michigan

1 Course • 6,638 learners

[View all 5 instructors](#)

Offered by

06

Closing slides

korniichuk	Filter by 'team.name' not 'possession_team.name'	0a92ce3 · 2 hours ago	5 Commits
sportmonks	Update to use Free Plan		5 hours ago
statsbomb	Filter by 'team.name' not 'possession_team.name'		2 hours ago
.gitignore	Initial commit		8 hours ago
LICENSE	Initial commit		8 hours ago
README.md	Initial commit		8 hours ago
requirements.txt	Initial commit		8 hours ago

BOOKMARKS

README

Unlicense license

football-analytics-pydata-warsaw

Football Analytics 101: Introduction to Football Analytics with Python

About

Football Analytics 101: Introduction to Football Analytics with Python

Readme

Unlicense license

Activity

0 stars

0 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Contributors



RUSLAN KORNIICHUK

Data and AI Platform Architect in Poland



Hire me

Solid experience of 11+ years in IT. Former Lead Engineer and Architect at Fortune 500 companies. Core areas: modern data platforms and cloud computing.

Public speaker at [European SME Congress](#), [Warsaw IT Days](#), [Data Science Summit](#), [Linux Autumn](#), and [EuroPython](#). Taught information system design and server-side technologies at the University of Silesia.

The honest, reliable, and hardworking person with always a keen eye for work details. **Click the button above to hire me**



DONATE TO UKRAINE ARMY FUNDRAISER



ARMY ASSISTANCE



SUPPORT THE FUND

 How will my money be used?

All funds raised to help the army will be spent exclusively on purchases for the Armed Forces of Ukraine and other components of the Defense Forces.

Payment method

PAYMENT BY CARD

TRANSFERS IN UKRAINE

SWIFT TRANSFERS

CRYPTOCURRENCY

Regularity

ONE TIME

SUBSCRIPTION

EVEN BETTER

Amount of donation*

€ 10

EUR, € ▼



+100 EUR



+200 EUR



+500 EUR



+1000 EUR

Email*

example@example.com

Do you want to receive emails from "Come Back Alive": analytics, news, and reports on how your support saves lives and strengthens the Ukrainian Defense Forces?*

☐ Agree ☐ Disagree

DONATE

tinyurl.com/PYDATA26

