

Package: ggshakeR (via r-universe)

October 19, 2026

Title Analytics and Visualization Package for Soccer Data

Version 0.2.0.9002

Description ggshakeR is an analysis and visualization R package that works with publically available soccer data. The datasets (for now) include FBref, StatsBomb, and understat.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports tidyr, magrittr, dplyr, ggplot2, ggsoccer, ggrepel, ggtext, TTR, stringi, purrr, ggforce, gridExtra

Suggests rmarkdown, knitr, testthat (>= 3.0.0), covr, stats

Config/testthat/edition 3

Depends R (>= 3.5.0)

URL <https://github.com/abhiyamishra/ggshakeR>,
<https://abhiyamishra.github.io/ggshakeR/>

BugReports <https://github.com/abhiyamishra/ggshakeR/issues>

VignetteBuilder knitr

Config/pak/sysreqs
libfontconfig1-dev libfreetype6-dev libicu-dev libjpeg-dev libpng-dev libxml2-dev libssl-dev

Repository <https://sportsdataverse.r-universe.dev>

Date/Publication 2023-08-18 14:21:11 UTC

RemoteUrl <https://github.com/abhiyamishra/ggshakeR>

RemoteRef HEAD

RemoteSha 51b7c2df7c691b3d386f463f18ffd0b152097a36

Contents

calculate_epv	2
calculate_threat	3
EPVGrid	4
plot_convexhull	4
plot_heatmap	5
plot_pass	6
plot_passflow	7
plot_passnet	8
plot_pizza	9
plot_scatter	10
plot_shot	12
plot_sonar	13
plot_timeline	13
plot_trendline	14
plot_voronoi	15
SampleEventData	16
SampleSBData	17
xTGrid	17
Index	18

calculate_epv	<i>Calculating EPV for passes, carries, etc</i>
---------------	---

Description

Calculating EPV for passes, carries, etc

Usage

calculate_epv(data, type = "opta")

Arguments

- | | |
|------|---|
| data | The dataframe that stores your data. Must contain starting x,y locations and ending x,y locations: x, y, finalX, finalY |
| type | indicator for what type of data the eventData. Currently, options include "opta" (default) and "statsbomb" |

Value

returns a dataframe object

Examples

```
## Not run:
endResult <- calculate_epv(test, type = "statsbomb")
endResult

## End(Not run)
```

calculate_threat	<i>Calculating xT for passes, carries, etc</i>
------------------	--

Description

Calculating xT for passes, carries, etc

Usage

```
calculate_threat(data, type = "opta")
```

Arguments

data	The dataframe that stores your data. Must contain starting x,y locations and ending x,y locations: x, y, finalX, finalY
type	indicator for what type of data the data. Currently, options include "opta" (default) and "statsbomb"

Value

returns a dataframe object

Examples

```
## Not run:
endResult <- calculate_threat(test, type = "statsbomb")
endResult

## End(Not run)
```

EPVGrid	<i>EPV calculation data for analysis</i>
---------	--

Description

a simple dataframe that is used for calculating expected possession values

Usage

EPVGrid

Format

A dataframe with 50 columns and 31 rows spanning a football pitch.

plot_convexhull	<i>Function for plotting convex hulls</i>
-----------------	---

Description

This function allows for data, that can be from Opta or StatsBomb, to be used for plotting convex hulls on top of an outline of a football pitch.

Usage

```
plot_convexhull(  
  data,  
  data_type = "statsbomb",  
  color = "#E74C3C",  
  title = "",  
  theme = "dark"  
)
```

Arguments

data	Data frame that houses pass data. Opta dataframe must contain atleast the following columns: x, y, finalX, finalY, playerId
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
color	The color of the outline of the convex hull
title	Title of the plot
theme	Indicates what theme the map must be shown in: dark (default), white, rose, almond

Value

a ggplot2 object

Examples

```
## Not run:
plot <- plot_convexhull(data = data, data_type = "opta", color = "blue", title = "Team 1")
plot

## End(Not run)
```

plot_heatmap	<i>Plotting heatmap</i>
--------------	-------------------------

Description

This function allows you to plot various types of heatmaps of starting x and y coordinates: hex binwidth heatmap, density heatmap, and binwidth heatmap

Usage

```
plot_heatmap(
  data,
  type = "",
  data_type = "statsbomb",
  binwidth = 20,
  theme = ""
)
```

Arguments

data	The dataframe that stores your data. Dataframe must contain atleast the following columns: x, y.
type	indicates the type of heatmap to plot. "hex" indicates hex bins, "density" indicates density (default), "binwidth" indicates binwidth heatmap pass, and "jdp" indicates a binned heatmap according jdp pitch markings.
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
binwidth	indicates the size of the bin width to construct heatmap for type "binwidth". The same argument name as the underlying call to geom_bin2d(). Default set to 20.
theme	indicates what theme the map must be shown in: dark (default), white, rose, almond

Value

returns a ggplot2 object

Examples

```
## Not run:
plot <- plot_heatmap(data = touchData, type = "hex")
plot

## End(Not run)
```

plot_pass

Plotting passes

Description

This function allows you to plot various types of plots that have passes as some sort of input. Data entered must have columns for which you want to plot with. Compatible with StatsBomb and Opta data.

Usage

```
plot_pass(
  data,
  data_type = "statsbomb",
  type = "sep",
  progressive_pass = FALSE,
  cross = FALSE,
  shot = FALSE,
  switch = FALSE,
  outcome = "all",
  theme = "dark"
)
```

Arguments

data	The data frame that stores your passing data. Opta data frame must contain at least the following columns: x, y, finalX, finalY
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
type	indicates the type of plot to pass. "sep" separates successful and unsuccessful passes. "all" plots all passes on one pitch. Default = "sep". Only available for StatsBomb data
progressive_pass	indicates whether to map out progressive passes. Only available for StatsBomb data
cross	indicates whether to map out crosses. Only available for StatsBomb data
shot	indicates whether to map out shot assists. Only available for StatsBomb data
switch	indicates whether to map out switches of play. Only available for StatsBomb data

outcome	indicates whether you want successful ("suc"), unsuccessful ("unsuc"), or all ("all"). Only available for StatsBomb data
theme	indicates what theme the map must be shown in: dark (default), white, rose, almond

Value

returns a ggplot2 object

Examples

```
## Not run:
plot <- plot_pass(data, type = "all", progressive_pass = TRUE)
plot

## End(Not run)
```

plot_passflow	<i>Passflow plot function</i>
---------------	-------------------------------

Description

This function takes in a dataframe and binsizes to make a passflow map. Compatible, for right now, with StatsBomb data only. Returns a ggplot object

Usage

```
plot_passflow(data, data_type = "statsbomb", binwidth = 0)
```

Arguments

data	Dataframe that must house pass data only and must contain atleast the following columns: x, y, finalX, finalY
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
binwidth	Details the bin size the passflow needs to bin to. The same argument name as the underlying call to geom_bin2d(). Default is 20.

Value

returns a ggplot2 object

Examples

```
## Not run:
plot <- plot_passflow(data, binwidth = 30)
plot

## End(Not run)
```

`plot_passnet`*Function for plotting pass networks*

Description

This function allows for data, that can be from Opta or Statsbomb, to be used for plotting pass networks.

Usage

```
plot_passnet(  
  data,  
  data_type = "statsbomb",  
  team_name,  
  scale_stat = "xT",  
  scale_color = "#E74C3C",  
  subtitle = "",  
  theme = "dark"  
)
```

Arguments

<code>data</code>	Data frame that houses pass data
<code>data_type</code>	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
<code>team_name</code>	The name of the team of which you want a pass network
<code>scale_stat</code>	Stat for the player node color scale. Choose between "xT" and "EPV"
<code>scale_color</code>	Color of higher end of xT/EPV color scale. Default set to "#E74C3C"
<code>subtitle</code>	Subtitle of the pass network plot
<code>theme</code>	The background theme -> "dark" or "light"

Value

a ggplot2 object

Examples

```
## Not run:  
plot <- plot_passnet(data, data_type = "opta", team_name = "Arsenal")  
plot  
  
## End(Not run)
```

plot_pizza	<i>Function for plotting percentile/pizza plots</i>
------------	---

Description

This function allows for data, that has to be scraped from FBref, to be used for plotting single and comparison percentile plots.

Usage

```
plot_pizza(
  data,
  type = "",
  template,
  color_possession = "#41ab5d",
  color_attack = "#2171b5",
  color_defense = "#fec44f",
  player_1,
  player_2,
  color_compare = "#41ab5d",
  season = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
  season_player_1 = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
  season_player_2 = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
  theme = ""
)
```

Arguments

data	Data frame can contain either one player or two depending on the type of plot made
type	Type of plot -> single and comparison
template	Selecting a group of pre-selected metrics for each position by position namely: outfielder, goalkeeper and custom
color_possession	Selecting the color for possession group of stats. To be used only for single player plot
color_attack	Selecting the color for attacking group of stats. To be used only for single player plot
color_defense	Selecting the color for defense group of stats. To be used only for single player plot
player_1	Selecting the first player. To be used only for comparison plot
player_2	Selecting the second player. To be used only for comparison plot
color_compare	Selecting the color of comparison to be used only for comparison plot
season	Specify what season to pick for a single player pizza chart. Pick the scouting period from the scouting period column in the data

season_player_1	Specify what season to pick for the first player in a pizza chart
season_player_2	Specify what season to pick for the second player in a pizza chart
theme	Specify the theme of the pizza chart -> dark, black, and white. Default set to dark

Details

For best image quality use: `ggsave("image.png", width = 2900, height = 2800, units = "px")`

Value

a ggplot2 object

Examples

```
## Not run:
plot1 <- plot_pizza(data = data, type = "comparison", template = "outfielder",
                    player_1 = "Nicolo Barella", player_2 = "Ilkay Gundogan",
                    season_player_1 = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
                    season_player_2 = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
                    color_compare = "lightgreen", theme = "black")

plot1

plot2 <- plot_pizza(data = data1, type = "single", template = "outfielder",
                    color_possession = "green", color_attack = "lightblue",
                    season = "Last 365 Days Men's Big 5 Leagues, UCL, UEL",
                    color_defense = "red", theme = "dark")

plot2

## End(Not run)
```

plot_scatter

Plotting scatter plots

Description

This function allows you to plot various types of plots that have passes as some sort of input. Compatible with any data frame of any data type. Returns a ggplot object.

Usage

```
plot_scatter(
  data,
  x = "",
  y = "",
  label = "",
```

```

    set_size_num = 5,
    set_size_var = "",
    set_color_num = "red",
    set_color_var = "",
    title = "",
    title_size = 25,
    subtitle = "",
    subtitle_size = 15,
    caption = "",
    caption_size = 10,
    theme = "classic"
)

```

Arguments

data	the dataframe passed in for plotting.
x	name of column name in data to be used on x-axis
y	name of column name in data to be used on y-axis
label	the name of column name in data to label the scatter plot
set_size_num	sets the size of the points set as a constant. Default size = 5.
set_size_var	Enter name of column name in data to set size based on variable.
set_color_num	sets the color of the points set as a constant. Can enter hexcode or a valid ggplot2 color. Default = "red"
set_color_var	Enter name of column name in data to set color based on variable.
title	pick the title of the scatter plot
title_size	sets the size of the title of the scatter plot. Default size = 25.
subtitle	pick the subtitle of the scatter plot
subtitle_size	sets the size of the subtitle of the scatter plot Default size = 15.
caption	pick the caption of the scatter plot
caption_size	sets the size of the caption of the scatter plot. Default size = 10.
theme	decide the theme of the plot between four choices: classic, minimal, grey, bw. Default = "classic"

Value

returns a ggplot2 object

Examples

```

## Not run:
plot <- plot_scatter(data, x = "player", y = "age", label = "team")
plot

## End(Not run)

```

plot_shot

Function for plotting shots

Description

This function allows for data, that has to be scraped from Understat, to be used for plotting shots.

Usage

```
plot_shot(
  data,
  type = "",
  bins = 30,
  highlight_goals = "",
  average_location = "",
  theme = ""
)
```

Arguments

data	Dataframe that houses shot data. Dataframe must contain atleast the following columns: X,Y,xG,result,name
type	Type of showcasing the shotmap: hexbin, density, point (default)
bins	Bin size for creating bins. Use this when using hexbin shotmap. The same argument name as the underlying call to geom_hex(). Default = 30.
highlight_goals	to choose to display only the goals in a different color.
average_location	for removing lines denoting average location of shots if need be.
theme	Theme preferences for display: dark (default), white, rose, almond

Value

a ggplot2 object

Examples

```
## Not run:
plot <- plot_shot(data, type = "hexbin", bins = 20,
  average_location = TRUE, highlight_goals = FALSE)

## End(Not run)
```

plot_sonar	<i>Function for plotting pass sonars</i>
------------	--

Description

This function allows for data, that can be from Opta or Statsbomb, to be used for plotting pass sonars.

Usage

```
plot_sonar(data, data_type = "statsbomb", title = "")
```

Arguments

data	Dataframe that houses pass data. Dataframe must contain atleast the following columns: x, y, finalX, finalY
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb"
title	Title of the passing sonar plot

Value

a ggplot2 object

Examples

```
## Not run:  
plot <- plot_sonar(data, data_type = "statsbomb")  
plot  
  
## End(Not run)
```

plot_timeline	<i>Create timeline charts (xG) using 'understat' event data.</i>
---------------	--

Description

This function allows to make match timelines using data collected event-by-event.

Usage

```
plot_timeline(
  data,
  match_year,
  team_home,
  team_away,
  color_home,
  color_away,
  theme = ""
)
```

Arguments

data	Data, for now only compatible with understat data.
match_year	the year the match was played.
team_home	home team according to data.
team_away	away team according to data.
color_home	line colour for the home team.
color_away	line colour for the away team.
theme	Choose from 4 ggplot2 themes -> dark, almond, rose, white.

Value

ggplot2 object of a xG timeline plot

Examples

```
## Not run:
plot <- plot_timeline(data = data, match_year = 2021,
                      team_home = "Manchester United", team_away = "Liverpool",
                      color_home = "#e31a1c", color_away = "#980043", theme = "dark")

plot

## End(Not run)
```

plot_trendline	<i>Function for plotting xG Trendline with FBref/ StatsBomb data.</i>
----------------	---

Description

The data can be scraped from FBref.
 Dataframe passed in must have the following column names:

Date (format: year-month-day).yyyy-mm-dd,
 Home_xG (xG for Home Team),
 Away_xG (xG for Away Team),
 Home (Home Team),
 Away (Away Team)

Usage

```
plot_trendline(data, team, color_xg, color_xga, rolling_average, theme = "")
```

Arguments

data	is for the dataset used. Select the number of matches wanted in the viz beforehand.
team	is to select the specific team for the viz. Team must be accurate as per FBref specifications.
color_xg	is for selecting color for xGoals.
color_xga	is for selecting the color for xGoalsAgainst.
rolling_average	is for setting the rolling average for the data.
theme	to select the theme from 4 options -> dark, almond, rose, white.

Details

For best clarity, export plot as a 2000x1000 png

Examples

```
## Not run:
plot <- plot_trendline(data = pl, team = "Tottenham",
                      color_xg = "#08519c", color_xga = "#cb181d",
                      rolling_average = 10, theme = "dark")

plot

## End(Not run)
```

plot_voronoi

Function for plotting voronoi diagrams on the pitch

Description

The function allows for creation of voronoi diagrams on a football pitch with either Opta or Statsbomb data.

Usage

```
plot_voronoi(
  data,
  data_type = "statsbomb",
  colour = "#E74C3C",
  fill = "",
  alpha = 0.4,
  title = "",
  theme = "dark"
)
```

Arguments

data	Dataframe that houses pass data. Opta/Statsbomb dataframe must contain atleast the following columns: x, y
data_type	Type of data that is being put in: opta or statsbomb. Default set to "statsbomb".
colour	The colour of the points in the voronoi plot.
fill	Name of column to add a fill component to the plot
alpha	Alpha value for opacity of fill. Default set to 0.4
title	Title of the plot.
theme	Indicates what theme the map must be shown in: dark (default), white, rose, almond.

Value

a ggplot2 object

Examples

```
## Not run:
plot <- plot_voronoi(data = data, colour = "blue", title = "Team 1")
plot

## End(Not run)
```

SampleEventData	<i>Sample event data frame</i>
-----------------	--------------------------------

Description

Event dataframe to play with and understand usage of functions

Usage

SampleEventData

Format

A dataframe with the columns x, y, finalX, finalY, minute, playerId, teamId, outcome and type.

SampleSBData	<i>Sample StatsBomb event data frame</i>
--------------	--

Description

StatsBomb event dataframe to play with and understand usage of functions. The following code was used with the use of StatsBombR:

```
Comp <- FreeCompetitions() %>% filter(competition_id == 11 & season_name == "2014/2015")
Matches <- FreeMatches(Comp) StatsBombData <- StatsBombFreeEvents(MatchesDF = Matches,
Parallel = TRUE) plotting_data <- allclean(StatsBombData) plotting_data <- plotting_data
%>% rename("x" = "location.x", "y" = "location.y", "finalX" = "pass.end_location.x",
"finalY" = "pass.end_location.y")
```

Usage

```
SampleSBData
```

Format

A dataframe with the columns x, y, finalX, finalY, and the rest of StatsBomb columns

xTGrid	<i>xT calculation data for analysis</i>
--------	---

Description

a simple dataframe that is used for calculating expected threat values

Usage

```
xTGrid
```

Format

A dataframe with 12 columns and 8 rows spanning a football pitch.

Index

* **datasets**

- EPVGrid, [4](#)
- SampleEventData, [16](#)
- SampleSBData, [17](#)
- xTGrid, [17](#)

calculate_epv, [2](#)
calculate_threat, [3](#)

EPVGrid, [4](#)

plot_convexhull, [4](#)
plot_heatmap, [5](#)
plot_pass, [6](#)
plot_passflow, [7](#)
plot_passnet, [8](#)
plot_pizza, [9](#)
plot_scatter, [10](#)
plot_shot, [12](#)
plot_sonar, [13](#)
plot_timeline, [13](#)
plot_trendline, [14](#)
plot_voronoi, [15](#)

SampleEventData, [16](#)
SampleSBData, [17](#)

xTGrid, [17](#)