

---

# **sportsreference Documentation**

***Release 0.1.0***

**Author**

**Oct 07, 2019**



---

## Contents:

---

<b>1</b>	<b>Examples</b>	<b>3</b>
1.1	Get instances of all NHL teams for the 2018 season . . . . .	3
1.2	Print every NBA team's name and abbreviation . . . . .	3
1.3	Get a specific NFL team's season information . . . . .	3
1.4	Print the date of every game for a NCAA Men's Basketball team . . . . .	4
1.5	Print the number of interceptions by the away team in a NCAA Football game . . . . .	4
1.6	Get a Pandas DataFrame of all stats for a MLB game . . . . .	4
1.6.1	API Documentation . . . . .	4
1.6.1.1	MLB Package . . . . .	4
1.6.1.2	NBA Package . . . . .	28
1.6.1.3	NCAAB Package . . . . .	49
1.6.1.4	NCAAF Package . . . . .	72
1.6.1.5	NFL Package . . . . .	93
1.6.1.6	NHL Package . . . . .	115
1.6.2	Examples . . . . .	133
1.6.2.1	Finding Tallest Players . . . . .	133
1.6.2.2	Writing To CSV and Pickle . . . . .	134
1.6.2.3	Finding Top Win Percentage By Year . . . . .	134
1.6.3	Installation . . . . .	134
1.6.4	Testing . . . . .	135
<b>2</b>	<b>Indices and tables</b>	<b>137</b>
	<b>Python Module Index</b>	<b>139</b>
	<b>Index</b>	<b>141</b>



Sportsreference is a free python API that pulls the stats from [www.sports-reference.com](http://www.sports-reference.com) and allows them to be easily be used in python-based applications, especially ones involving data analytics and machine learning.

Sportsreference exposes a plethora of sports information from major sports leagues in North America, such as the MLB, NBA, College Football and Basketball, NFL, and NHL. Every sport has its own set of valid API queries ranging from the list of teams in a league, to the date and time of a game, to the total number of wins a team has secured during the season, and many, many more metrics that paint a more detailed picture of how a team has performed during a game or throughout a season.



The following are a few examples showcasing how easy it can be to collect an abundance of metrics and information from all of the tracked leagues. The examples below are only a miniscule subset of the total number of statistics that can be pulled using sportsreference. Visit the documentation on [Read The Docs](#) for a complete list of all information exposed by the API.

### 1.1 Get instances of all NHL teams for the 2018 season

```
from sportsreference.nhl.teams import Teams

teams = Teams(2018)
```

### 1.2 Print every NBA team's name and abbreviation

```
from sportsreference.nba.teams import Teams

teams = Teams()
for team in teams:
    print(team.name, team.abbreviation)
```

### 1.3 Get a specific NFL team's season information

```
from sportsreference.nfl.teams import Teams

teams = Teams()
lions = teams('DET')
```

## 1.4 Print the date of every game for a NCAA Men's Basketball team

```
from sportsreference.ncaab.schedule import Schedule

purdue_schedule = Schedule('purdue')
for game in purdue_schedule:
    print(game.date)
```

## 1.5 Print the number of interceptions by the away team in a NCAA Football game

```
from sportsreference.ncaaf.boxscore import Boxscore

championship_game = Boxscore('2018-01-08-georgia')
print(championship_game.away_interceptions)
```

## 1.6 Get a Pandas DataFrame of all stats for a MLB game

```
from sportsreference.mlb.boxscore import Boxscore

game = Boxscore('BOS201806070')
df = game.dataframe
```

### 1.6.1 API Documentation

#### 1.6.1.1 MLB Package

The MLB package offers multiple modules which can be used to retrieve information and statistics for Major League Baseball, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of runs scored to the number of sacrifice flies, to the slugging percentage and much more. The Boxscore can be easily queried by passing a boxscore's URI on sports-reference.com which can be retrieved from the Schedule class (see Schedule module below for more information on retrieving game-specific information).

```
from sportsreference.mlb.boxscore import Boxscore

game_data = Boxscore('BOS/BOS201808020')
print(game_data.home_runs)    # Prints 15
print(game_data.away_runs)    # Prints 7
df = game_data.dataframe      # Returns a Pandas DataFrame of game metrics
```

The Boxscore module also contains a Boxscores class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.



```
from datetime import datetime
from sportsreference.mlb.boxscore import Boxscores

games_today = Boxscores(datetime.today())
print(games_today.games)  # Prints a dictionary of all matchups for today
```

The Boxscores class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```
from datetime import datetime
from sportsreference.mlb.boxscore import Boxscores

# Pulls all games between and including July 17, 2017 and July 20, 2017
games = Boxscores(datetime(2017, 7, 17), datetime(2017, 7, 20))
# Prints a dictionary of all results from July 17, 2017 and July 20, 2017
print(games.games)
```

```
class sportsreference.mlb.boxscore.Boxscore(uri)
```

Bases: object

Detailed information about the final statistics for a game.

Stores all relevant information for a game such as the date, time, location, result, and more advanced metrics such as the number of strikes, a pitcher's influence on the game, the number of putouts and much more.

**Parameters** `uri` (*string*) – The relative link to the boxscore HTML page, such as 'BOS/BOS201806070'.

**attendance**

Returns an `int` of the game's listed attendance.

**away\_assists**

Returns an `int` of the number of assists the away team registered.

**away\_at\_bats**

Returns an `int` of the number of at bats the away team had.

**away\_average\_leverage\_index**

Returns a `float` of the amount of pressure the away team's pitcher faced during the game. 1.0 denotes average pressure while numbers less than 0 denote lighter pressure.

**away\_base\_out\_runs\_added**

Returns a `float` of the number of base out runs added by the away team.

**away\_base\_out\_runs\_saved**

Returns a `float` of the number of runs saved by the away pitcher based on the number of players on bases. 0.0 denotes an average value.

**away\_bases\_on\_balls**

Returns an `int` of the number of bases the away team registered as a result of balls.

**away\_batting\_average**

Returns a `float` of the batting average for the away team.

**away\_earned\_runs**

Returns a `float` of the number of runs the away team earned.

**away\_fly\_balls**

Returns an `int` of the number of fly balls the away team allowed.

**away\_game\_score**

Returns an `int` of the starting away pitcher's score determine by many factors, such as number of runs scored against, number of strikes, etc.

**away\_grounded\_balls**

Returns an `int` of the number of grounded balls the away team allowed.

**away\_hits**

Returns an `int` of the number of hits the away team had.

**away\_home\_runs**

Returns an `int` of the number of times the away team gave up a home run.

**away\_inherited\_runners**

Returns an `int` of the number of runners a pitcher inherited when he entered the game.

**away\_inherited\_score**

Returns an `int` of the number of scorers a pitcher inherited when he entered the game.

**away\_innings\_pitched**

Returns a `float` of the number of innings the away team pitched.

**away\_line\_drives**

Returns an `int` of the number of line drives the away team allowed.

**away\_on\_base\_percentage**

Returns a `float` of the percentage of at bats that result in the batter getting on base.

**away\_on\_base\_plus**

Returns a `float` of the on base percentage plus the slugging percentage. Percentage ranges from 0-1.

**away\_pitches**

Returns an `int` of the number of pitches the away team faced.

**away\_plate\_appearances**

Returns an `int` of the number of plate appearances the away team made.

**away\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the away team.

**away\_putouts**

Returns an `int` of the number of putouts the away team registered.

**away\_rbi**

Returns an `int` of the number of runs batted in the away team registered.

**away\_runs**

Returns an `int` of the number of runs the away team scored.

**away\_slugging\_percentage**

Returns a `float` of the slugging percentage for the away team based on the number of bases gained per at-bat with bigger plays getting more weight.

**away\_strikeouts**

Returns an `int` of the number of times the away team was struck out.

**away\_strikes**

Returns an `int` of the number of times a strike was called against the away team.

**away\_strikes\_by\_contact**

Returns an `int` of the number of times the away team struck out a batter who made contact with the pitch.

**away\_strikes\_looking**

Returns an `int` of the number of times the away team struck out a batter who was looking.

**away\_strikes\_swinging**

Returns an `int` of the number of times the away team struck out a batter who was swinging.

**away\_unknown\_bat\_type**

Returns an `int` of the number of away at bats that were not properly tracked and therefore cannot be safely placed in another statistical category.

**away\_win\_probability\_added**

Returns a `float` of the total positive influence the away team's offense had on the outcome of the game.

**away\_win\_probability\_by\_pitcher**

Returns a `float` of the amount of influence the away pitcher had on the game's result with 0.0 denoting zero influence and 1.0 denoting he was solely responsible for the team's win.

**away\_win\_probability\_for\_offensive\_player**

Returns a `float` of the overall influence the away team's offense had on the outcome of the game where 0.0 denotes no influence and 1.0 denotes the offense was solely responsible for the outcome.

**away\_win\_probability\_subtracted**

Returns a `float` of the total negative influence the away team's offense had on the outcome of the game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string URI that is used to instantiate the class, such as 'BOS201806070'.

**date**

Returns a `string` of the date the game took place.

**duration**

MM'.

**Type** Returns a `string` of the game's duration in the format 'H

**home\_assists**

Returns an `int` of the number of assists the home team registered.

**home\_at\_bats**

Returns an `int` of the number of at bats the home team had.

**home\_average\_leverage\_index**

Returns a `float` of the amount of pressure the home team's pitcher faced during the game. 1.0 denotes average pressure while numbers less than 0 denote lighter pressure.

**home\_base\_out\_runs\_added**

Returns a `float` of the number of base out runs added by the home team.

**home\_base\_out\_runs\_saved**

Returns a `float` of the number of runs saved by the home pitcher based on the number of players on bases. 0.0 denotes an average value.

**home\_bases\_on\_balls**

Returns an `int` of the number of bases the home team registered as a result of balls.

**home\_batting\_average**

Returns a `float` of the batting average for the home team.

**home\_earned\_runs**

Returns a `float` of the number of runs the home team earned.

**home\_fly\_balls**

Returns an `int` of the number of fly balls the home team allowed.

**home\_game\_score**

Returns an `int` of the starting home pitcher's score determine by many factors, such as number of runs scored against, number of strikes, etc.

**home\_grounded\_balls**

Returns an `int` of the number of grounded balls the home team allowed.

**home\_hits**

Returns an `int` of the number of hits the home team had.

**home\_home\_runs**

Returns an `int` of the number of times the home team gave up a home run.

**home\_inherited\_runners**

Returns an `int` of the number of runners a pitcher inherited when he entered the game.

**home\_inherited\_score**

Returns an `int` of the number of scorers a pitcher inherited when he entered the game.

**home\_innings\_pitched**

Returns a `float` of the number of innings the home team pitched.

**home\_line\_drives**

Returns an `int` of the number of line drives the home team allowed.

**home\_on\_base\_percentage**

Returns a `float` of the percentage of at bats that result in the batter getting on base.

**home\_on\_base\_plus**

Returns a `float` of the on base percentage plus the slugging percentage. Percentage ranges from 0-1.

**home\_pitches**

Returns an `int` of the number of pitches the home team faced.

**home\_plate\_appearances**

Returns an `int` of the number of plate appearances the home team made.

**home\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the home team.

**home\_putouts**

Returns an `int` of the number of putouts the home team registered.

**home\_rbi**

Returns an `int` of the number of runs batted in the home team registered.

**home\_runs**

Returns an `int` of the number of runs the home team scored.

**home\_slugging\_percentage**

Returns a `float` of the slugging percentage for the home team based on the number of bases gained per at-bat with bigger plays getting more weight.

**home\_strikeouts**

Returns an `int` of the number of times the home team was struck out.

**home\_strikes**

Returns an `int` of the number of times a strike was called against the home team.

**home\_strikes\_by\_contact**

Returns an `int` of the number of times the home team struck out a batter who made contact with the pitch.

**home\_strikes\_looking**

Returns an `int` of the number of times the home team struck out a batter who was looking.

**home\_strikes\_swinging**

Returns an `int` of the number of times the home team struck out a batter who was swinging.

**home\_unknown\_bat\_type**

Returns an `int` of the number of home at bats that were not properly tracked and therefore cannot be safely placed in another statistical category.

**home\_win\_probability\_added**

Returns a `float` of the total positive influence the home team's offense had on the outcome of the game.

**home\_win\_probability\_by\_pitcher**

Returns a `float` of the amount of influence the home pitcher had on the game's result with 0.0 denoting zero influence and 1.0 denoting he was solely responsible for the team's win.

**home\_win\_probability\_for\_offensive\_player**

Returns a `float` of the overall influence the home team's offense had on the outcome of the game where 0.0 denotes no influence and 1.0 denotes the offense was solely responsible for the outcome.

**home\_win\_probability\_subtracted**

Returns a `float` of the total negative influence the home team's offense had on the outcome of the game.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'LAD' for the Los Angeles Dodgers.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Los Angeles Dodgers'.

**time**

Returns a `string` of the time the game started.

**time\_of\_day**

Returns a `string` constant indicated whether the game was played during the day or at night.

**venue**

Returns a `string` of the name of the ballpark where the game was played.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'HOU' for the Houston Astros.

**winning\_name**

Returns a `string` of the winning team's name, such as 'Houston Astros'.

**class** sportsreference.mlb.boxscore.BoxscorePlayer(*player\_id*, *player\_name*,  
*player\_data*)

Bases: `sportsreference.mlb.player.AbstractPlayer`

Get player stats for an individual game.

Given a player ID, such as 'altuvjo01' for Jose Altuve, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

**Parameters**

- **player\_id**(*string*) – A player’s ID according to baseball-reference.com, such as ‘altuvjo01’ for Jose Altuve. The player ID can be found by navigating to the player’s stats page and getting the string between the final slash and the ‘.html’ in the URL. In general, the ID is in the format ‘LLLLLFFNN’ where ‘LLLLL’ are the first 5 letters in the player’s last name, ‘FF’, are the first 2 letters in the player’s first name, and ‘NN’ is a number starting at ‘01’ for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name**(*string*) – A string representing the player’s first and last name, such as ‘Jose Altuve’.
- **player\_data**(*string*) – A string representation of the player’s HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**average\_leverage\_index**

Returns a `float` of the amount of pressure the player faced during the game. 1.0 denotes average pressure while numbers less than 0 denote lighter pressure.

**average\_leverage\_index\_pitcher**

Returns a `float` of the amount of pressure the pitcher faced during the game. 1.0 denotes average pressure while numbers less than 0 denote lighter pressure.

**base\_out\_runs\_added**

Returns a `float` of the number of base out runs added by the player.

**base\_out\_runs\_saved**

Returns a `float` of the number of runs saved by the pitcher based on the number of players on bases. 0.0 denotes an average value.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values for the specified game.

**earned\_runs\_against**

Returns a `float` of the player’s overall Earned Runs Against average as calculated by  $9 * \text{earned\_runs} / \text{innings\_pitched}$ .

**fly\_balls**

Returns an `int` of the number of fly balls the player allowed.

**game\_score**

Returns an `int` of the pitcher’s score determine by many factors, such as number of runs scored against, number of strikes, etc.

**grounded\_balls**

Returns an `int` of the number of grounded balls the player allowed.

**home\_runs\_thrown**

Returns an `int` of the number of home runs the player threw.

**inherited\_runners**

Returns an `int` of the number of runners a relief pitcher inherited.

**inherited\_score**

Returns an `int` of the number of runners on base when a relief pitcher entered the game that ended up scoring.

**innings\_pitched**

Returns an `int` of the number of innings the player pitched in.

**line\_drives**

Returns an `int` of the number of line drives the player allowed.

**pitches\_thrown**

Returns an `int` of the number of pitches the player threw.

**strikes**

Returns an `int` of the number of times a strike was called against the player.

**strikes\_contact**

Returns an `int` of the number of times the player threw a strike when the player made contact with the ball.

**strikes\_looking**

Returns an `int` of the number of times the player threw a strike with the player looking.

**strikes\_swinging**

Returns an `int` of the number of times the player threw a strike with the batter swinging.

**strikes\_thrown**

Returns an `int` of the number of times a strikes the player threw.

**unknown\_bat\_types**

Returns an `int` of the number of line drives the player allowed.

**win\_probability\_added**

Returns a `float` of the total positive influence the player's offense had on the outcome of the game.

**win\_probability\_added\_pitcher**

Returns a `float` of the total positive influence the pitcher's offense had on the outcome of the game.

**win\_probability\_for\_offensive\_player**

Returns a `float` of the overall influence the player's offense had on the outcome of the game where 0.0 denotes no influence and 1.0 denotes the offense was solely responsible for the outcome.

**win\_probability\_subtracted**

Returns a `float` of the total negative influence the player's offense had on the outcome of the game.

**class** sportsreference.mlb.boxscore.Boxscores(*date*, *end\_date=None*)

Bases: `object`

Search for MLB games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

**Parameters**

- **date** (*datetime object*) – The date to search for any matches. The month, day, and year are required for the search, but time is not factored into the search.
- **end\_date** (*datetime object (optional)*) – Optionally specify an end date to iterate until. All boxscores starting from the date specified in the 'date' parameter up to and including the boxscores specified in the 'end\_date' parameter will be pulled. If left empty, or if 'end\_date' is prior to 'date', only the games from the day specified in the 'date' parameter will be saved.

**games**

Returns a `dictionary` object representing all of the games played on the requested day. Dictionary is in the following format:

```
{
  'date': [ # 'date' is the string date in format 'MM-DD-YYYY'
    {
```

(continues on next page)

(continued from previous page)

```

        'home_name': Name of the home team, such as 'New York
                      Yankees' (`str`),
        'home_abbr': Abbreviation for the home team, such as
                      'NYY' (`str`),
        'away_name': Name of the away team, such as 'Houston
                      Astros' (`str`),
        'away_abbr': Abbreviation for the away team, such as
                      'HOU' (`str`),
        'boxscore': String representing the boxscore URI, such
                      as 'SLN/SLN201807280' (`str`),
        'winning_name': Full name of the winning team, such as
                      'New York Yankees' (`str`),
        'winning_abbr': Abbreviation for the winning team, such
                      as 'NYY' (`str`),
        'losing_name': Full name of the losing team, such as
                      'Houston Astros' (`str`),
        'losing_abbr': Abbreviation for the losing team, such
                      as 'HOU' (`str`),
        'home_score': Integer score for the home team (`int`),
        'away_score': Integer score for the away team (`int`)
    },
    { ... },
    ...
]
}

```

If no games were played on ‘date’, the list for [‘date’] will be empty.

## Player

The Player module contains an abstract base class that can be inherited by both the `BoxscorePlayer` and `Player` classes in the `Boxscore` and `Roster` modules, respectively. All of the properties that appear in the `AbstractPlayer` class can be read from either of the two child classes mentioned above.

**class** `sportsreference.mlb.player.AbstractPlayer` (*player\_id, player\_name, player\_data*)

Bases: `object`

Get player information and stats for all seasons.

Given a player ID, such as ‘altuvjo01’ for Jose Altuve, capture all relevant stats and information like name, nationality, height/weight, career home runs, last season’s batting average, salary, contract amount, and much more.

By default, the class instance will return the player’s career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [baseball-reference.com](http://baseball-reference.com).

**Parameters** `player_id` (*string*) – A player’s ID according to [basketball-reference.com](http://basketball-reference.com), such as ‘altuvjo01’ for Jose Altuve. The player ID can be found by navigating to the player’s stats page and getting the string between the final slash and the ‘.html’ in the URL. In general, the ID is in the format ‘LLLLLFFNN’ where ‘LLLLL’ are the first 5 letters in the player’s last name, ‘FF’, are the first 2 letters in the player’s first name, and ‘NN’ is a number starting at ‘01’ for the first time that player ID has been used and increments by 1 for every successive player.

**assists**

Returns an `int` of the number of assists the player had.



**at\_bats**

Returns an `int` of the number of at bats the player had.

**bases\_on\_balls**

Returns an `int` of the number of bases the player registered as a result of balls.

**bases\_on\_balls\_given**

Returns an `int` of the number of bases on balls the player has given as a pitcher.

**batters\_faced**

Returns an `int` of the number of batters the pitcher has faced.

**batting\_average**

Returns a `float` of the batting average for the player.

**earned\_runs\_allowed**

Returns an `int` of the number of earned runs the player allowed as a pitcher.

**hits**

Returns an `int` of the number of hits the player had.

**hits\_allowed**

Returns an `int` of the number of hits the player allowed as a pitcher.

**name**

Returns a `string` of the player's name, such as 'Jose Altuve'.

**on\_base\_percentage**

Returns a `float` of the percentage of at bats that result in the batter getting on base.

**on\_base\_plus\_slugging\_percentage**

Returns a `float` of the on base percentage plus the slugging percentage. Percentage ranges from 0-1.

**plate\_appearances**

Returns an `int` of the number of plate appearances the player had.

**player\_id**

Returns a `string` of the player's ID on sports-reference, such as 'altuvjo01' for Jose Altuve.

**putouts**

Returns an `int` of the number of putouts the player had.

**runs**

Returns an `int` of the number of runs the player scored.

**runs\_allowed**

Returns an `int` of the number of runs the player allowed as a pitcher.

**runs\_batted\_in**

Returns an `int` of the number of runs batted in the player registered.

**slugging\_percentage**

Returns a `float` of the slugging percentage for the player based on the number of bases gained per at-bat with bigger plays getting more weight.

**strikeouts**

Returns an `int` of the number of strikeouts the player threw as a pitcher.

**times\_struck\_out**

Returns an `int` of the number of times the player was struck out.

## Roster

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the `Player` class which has detailed information ranging from career home runs to single-season stats and player height, weight, and nationality. The following is an example on collecting career information for José Altuve.

```
from sportsreference.mlb.roster import Player

altuve = Player('altuvjo01')
print(altuve.name)    # Prints 'José Altuve'
print(altuve.hits)    # Prints Altuve's career hits total
# Prints a Pandas DataFrame of all relevant stats per season for Altuve
print(altuve.dataframe)
```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific season, call the class instance with the season string. All future property requests will return the season-specific stats.

```
from sportsreference.mlb.roster import Player

altuve = Player('altuvjo01') # Currently pulling career stats
print(altuve.hits)          # Prints Altuve's CAREER hits total
# Prints Altuve's hits total only for the 2017 season
print(altuve('2017').hits)
# Prints Altuve's home runs total for the 2017 season only
print(altuve.home_runs)
```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.mlb.roster import Player

altuve = Player('altuvjo01') # Currently pulling career stats
# Prints Altuve's hits total only for the 2017 season
print(altuve('2017').hits)
print(altuve('Career').hits) # Prints Altuve's career hits total
```

In addition, the Roster module also contains the `Roster` class which can be used to pull all players on a team's roster during a given season and creates instances of the `Player` class for each team member and adds them to a list to be easily queried.

```
from sportsreference.mlb.roster import Roster

astros = Roster('HOU')
for player in astros.players:
    # Prints the name of all players who played for the Astros in the most
    # recent season.
    print(player.name)
```

```
class sportsreference.mlb.roster.Player(player_id)
    Bases: sportsreference.mlb.player.AbstractPlayer
```

Get player information and stats for all seasons.

Given a player ID, such as 'altuvjo01' for Jose Altuve, capture all relevant stats and information like name, nationality, height/weight, career home runs, last season's batting average, salary, contract amount, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [baseball-reference.com](http://baseball-reference.com).

**Parameters** `player_id` (*string*) – A player's ID according to [basketball-reference.com](http://basketball-reference.com), such as 'altuvjo01' for Jose Altuve. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'LLLLLFFNN' where 'LLLLL' are the first 5 letters in the player's last name, 'FF', are the first 2 letters in the player's first name, and 'NN' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.

**assists**

Returns an `int` of the number of assists the player had.

**at\_bats**

Returns an `int` of the number of at bats the player had.

**balks**

Returns an `int` of the number of times the pitcher balked.

**bases\_on\_balls**

Returns an `int` of the number of bases the player registered as a result of balls.

**bases\_on\_balls\_given**

Returns an `int` of the number of bases on balls the player has given as a pitcher.

**bases\_on\_balls\_given\_per\_nine\_innings**

Returns a `float` of the number of bases on balls the pitcher has given per nine innings played.

**batters\_struckout\_per\_nine\_innings**

Returns a `float` of the number of batters the pitcher has struck out per nine innings played.

**batting\_average**

Returns a `float` of the batting average for the player.

**birth\_date**

Returns a `datetime` object of the day and year the player was born.

**complete\_games**

Returns an `int` of the number of complete games the player has participated in.

**contract**

Returns a `dictionary` of the player's contract where each key is a `string` of the year, such as '2017' and each value is a `dictionary` with the `string` key-value pairs of the player's age, team name, and salary.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values where each index is a different season plus the career stats.

**defensive\_chances**

Returns an `int` of the number of defensive chances (equal to the number of putouts + assists + errors) the player had.

**defensive\_runs\_saved\_above\_average**

Returns an `int` of the number of defensive runs the player saved compared to an average player.

**defensive\_runs\_saved\_above\_average\_per\_innings**

Returns an `int` of the number of defensive runs the player was worth per 1,200 innings compared to an average player.

**double\_plays\_turned**

Returns an `int` of the number of double plays the player was involved in.

**doubles**

Returns an `int` of the number of doubles the player hit.

**earned\_runs\_allowed**

Returns an `int` of the number of earned runs the player allowed as a pitcher.

**era**

Returns a `float` of the pitcher's Earned Runs Average.

**era\_plus**

Returns a `float` of the pitcher's ERA while adjusted for the ballpark.

**errors**

Returns an `int` of the number of errors the player made.

**fielding\_independent\_pitching**

Returns a `float` of the pitcher's effectiveness at preventing home runs, bases on balls, and hitting players with pitches, while causing strikeouts.

**fielding\_percentage**

Returns a `float` of the players fielding percentage, equivalent to  $(\text{putouts} + \text{assists}) / (\text{putouts} + \text{assists} + \text{errors})$ . Percentage ranges from 0-1.

**games**

Returns an `int` of the number of games the player participated in.

**games\_catcher**

Returns an `int` of the number of games the player was in the lineup as a catcher.

**games\_center\_fielder**

Returns an `int` of the number of games the player was in the lineup as a center fielder.

**games\_designated\_hitter**

Returns an `int` of the number of games the player was in the lineup as a designated hitter.

**games\_finished**

Returns an `int` of the number of games the player finished as a pitcher.

**games\_first\_baseman**

Returns an `int` of the number of games the player was in the lineup as a first baseman.

**games\_in\_batting\_order**

Returns an `int` of the number of games the player was in the batting lineup.

**games\_in\_defensive\_lineup**

Returns an `int` of the number of games the player was in the defensive lineup.

**games\_left\_fielder**

Returns an `int` of the number of games the player was in the lineup as a left fielder.

**games\_outfielder**

Returns an `int` of the number of games the player was in the lineup as an outfielder.

**games\_pinch\_hitter**

Returns an `int` of the number of games the player was in the lineup as a pinch hitter.

**games\_pinch\_runner**

Returns an `int` of the number of games the player was in the lineup as a pinch runner.

**games\_pitcher**

Returns an `int` of the number of games the player was in the lineup as a pitcher.

**games\_right\_fielder**

Returns an `int` of the number of games the player was in the lineup as a right fielder.

**games\_second\_baseman**

Returns an `int` of the number of games the player was in the lineup as a second baseman.

**games\_shortstop**

Returns an `int` of the number of games the player was in the lineup as a shortstop.

**games\_started**

Returns an `int` of the number of games the player started.

**games\_third\_baseman**

Returns an `int` of the number of games the player was in the lineup as a third baseman.

**grounded\_into\_double\_plays**

Returns an `int` of the number of double plays the player grounded into.

**height**

Returns a `string` of the players height in the format “feet-inches”.

**hits**

Returns an `int` of the number of hits the player had.

**hits\_against\_per\_nine\_innings**

Returns a `float` of the number of hits the player has given per nine innings played.

**hits\_allowed**

Returns an `int` of the number of hits the player allowed as a pitcher.

**home\_runs**

Returns an `int` of the number of home runs the player hit.

**home\_runs\_against\_per\_nine\_innings**

Returns a `float` of the number of home runs the pitcher has given per nine innings played.

**home\_runs\_allowed**

Returns an `int` of the number of home runs a player has allowed as a pitcher.

**innings\_played**

Returns a `float` of the total number of innings the player has played in.

**intentional\_bases\_on\_balls**

Returns an `int` of the number of times the player has been intentionally walked by the opposition.

**intentional\_bases\_on\_balls\_given**

Returns an `int` of the number of bases the player has intentionally given as a pitcher.

**league\_fielding\_percentage**

Returns a `float` of the average fielding percentage for the league at the player’s position. Percentage ranges from 0-1.

**league\_range\_factor\_per\_game**

Returns a `float` of the average range factor for the league per game, equal to (putouts + assists) / games\_played.

**league\_range\_factor\_per\_nine\_innings**

Returns a `float` of the average range factor for the league per nine innings, equal to 9 \* (putouts + assists) / innings\_played.

**losses**

Returns an `int` of the number of games the player has lost as a pitcher.

**name**

Returns a `string` of the player’s name, such as ‘Jose Altuve’.

**nationality**

Returns a `string` constant denoting which country the player originates from.

**on\_base\_percentage**

Returns a `float` of the percentage of at bats that result in the batter getting on base.

**on\_base\_plus\_slugging\_percentage**

Returns a `float` of the on base percentage plus the slugging percentage. Percentage ranges from 0-1.

**on\_base\_plus\_slugging\_percentage\_plus**

Returns an `int` of the on base percentage plus the slugging percentage, adjusted to the player's ballpark.

**plate\_appearances**

Returns an `int` of the number of plate appearances the player had.

**position**

Returns a `string` constant of the player's primary position.

**putouts**

Returns an `int` of the number of putouts the player had.

**range\_factor\_per\_game**

Returns a `float` of the players range factor per game, equal to  $9 * (\text{putouts} + \text{assists}) / \text{games\_played}$ .

**range\_factor\_per\_nine\_innings**

Returns a `float` of the players range factor per nine innings, equal to  $9 * (\text{putouts} + \text{assists}) / \text{innings\_played}$ .

**runs**

Returns an `int` of the number of runs the player scored.

**runs\_allowed**

Returns an `int` of the number of runs the player allowed as a pitcher.

**runs\_batted\_in**

Returns an `int` of the number of runs batted in the player registered.

**sacrifice\_flies**

Returns an `int` of the number of sacrifice flies the player hit.

**sacrifice\_hits**

Returns an `int` of the number of sacrifice hits or sacrifice bunts the player made.

**saves**

Returns an `int` of the number of saves the player made as a pitcher.

**season**

Returns a `string` of the season in the format 'YYYY', such as '2017'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**shutouts**

Returns an `int` of the number of times the player did not allow any runs and threw a complete game as a pitcher.

**slugging\_percentage**

Returns a `float` of the slugging percentage for the player based on the number of bases gained per at-bat with bigger plays getting more weight.

**stolen\_bases**

Returns an `int` of the number of bases the player has stolen.

**strikeouts**

Returns an `int` of the number of strikeouts the player threw as a pitcher.

**strikeouts\_thrown\_per\_walk**

Returns a `float` of the number of batters the pitcher has struck out per the number of walks given.

**team\_abbreviation**

Returns a `string` of the team's abbreviation, such as 'HOU' for the Houston Astros.

**times\_caught\_stealing**

Returns an `int` of the number of times the player was caught stealing.

**times\_hit\_by\_pitch**

Returns an `int` of the number of times the player has been hit by a pitch.

**times\_hit\_player**

Returns an `int` of the number of times the pitcher hit a player with a pitch.

**times\_struck\_out**

Returns an `int` of the number of times the player was struck out.

**total\_bases**

Returns an `int` of the number of bases the player has gained.

**total\_fielding\_runs\_above\_average**

Returns an `int` of the number of runs the player was worth compared to an average player.

**total\_fielding\_runs\_above\_average\_per\_innings**

Returns an `int` of the number of runs the player was worth per 1,200 innings compared to an average player.

**triples**

Returns an `int` of the number of triples the player hit.

**weight**

Returns an `int` of the player's weight in pounds.

**whip**

Returns a `float` of the pitcher's WHIP score, equivalent to (bases on balls + hits) / innings played.

**wild\_pitches**

Returns an `int` of the number of wild pitches the player has thrown.

**win\_percentage**

Returns a `float` of the players winning percentage as a pitcher. Percentage ranges from 0-1.

**wins**

Returns an `int` of the number of games the player has won as a pitcher.

**class** sportsreference.mlb.roster.**Roster** (*team*, *year=None*, *slim=False*)

Bases: `object`

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the `Player` class for each player, containing a detailed list of the players statistics and information.

**Parameters**

- **team** (*string*) – The team's abbreviation, such as 'HOU' for the Houston Astros.
- **year** (*string (optional)*) – The 4-digit year to pull the roster from, such as '2018'. If left blank, defaults to the most recent season.
- **slim** (*boolean (optional)*) – Set to `True` to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective

stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to False.

**players**

Returns a list of player instances for each player on the requested team's roster if the `slim` property is False when calling the Roster class. If the `slim` property is True, returns a dictionary where each key is a string of the player's ID and each value is the player's first and last name as listed on the roster page.

## Schedule

The Schedule module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.mlb.schedule import Schedule

houston_schedule = Schedule('HOU')
for game in houston_schedule:
    print(game.date) # Prints the date the game was played
    print(game.result) # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** sportsreference.mlb.schedule.**Game**(*game\_data*, *year*)

Bases: object

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

**Parameters**

- **game\_data** (*string*) – The row containing the specified game information.
- **year** (*string*) – The year of the current season.

**attendance**

Returns an int of the total listed attendance for the game.

**boxscore**

Returns an instance of the `Boxscore` class containing more detailed stats on the game.

**boxscore\_index**

Returns a string of the URI for a boxscore which can be used to access or index a game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame representing the `Boxscore` class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property. The index for the DataFrame is the boxscore string.

**date**

Returns a string of the date the game was played on.

**datetime**

Returns a datetime object of the month, day, year, and time the game was played.



**day\_or\_night**

Returns a `string` constant to indicate whether the game was played during the day or night.

**game**

Returns an `int` of the game in the season, where 1 is the first game of the season.

**game\_duration**

MM'.

**Type** Returns a `string` of the game's total duration in the format 'H

**game\_number\_for\_day**

Returns an `int` denoting which game is played for the team during the given day. Default value is 1 where a team plays only one game during the day, but can be higher for double headers, etc. For example, if a team has a double header one day, the first game of the day will return 1 while the second game will return 2.

**games\_behind**

Returns a `float` of the number of games behind the leader the team is. 0.0 indicates the team is tied for first. Negative numbers indicate the number of games a team is ahead of the second place team.

**innings**

Returns an `int` of the total number of innings that were played.

**location**

Returns a `string` constant to indicate whether the game was played at home or away.

**loser**

Returns a `string` of the name of the losing pitcher.

**opponent\_abbr**

Returns a `string` of the opponent's 3-letter abbreviation, such as 'NYY' for the New York Yankees.

**rank**

Returns an `int` of the team's rank in the league with 1 being the best team.

**record**

Returns a `string` of the team's record in the format 'W-L'.

**result**

Returns a `string` constant to indicate whether the team won or lost.

**runs\_allowed**

Returns an `int` of the total number of runs that the team allowed.

**runs\_scored**

Returns an `int` of the total number of runs that were scored by the team.

**save**

Returns a `string` of the name of the pitcher credited with the save if applicable. If no saves, returns `None`.

**streak**

Returns a `string` of the team's winning/losing streak at the conclusion of the requested game. A winning streak is denoted by a number of '+' signs for the number of consecutive wins and a losing streak is denoted by a '-' sign.

**winner**

Returns a `string` of the name of the winning pitcher.

**class** sportsreference.mlb.schedule.**Schedule** (*abbreviation*, *year=None*)

Bases: `object`

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

#### Parameters

- **abbreviation** (*string*) – A team's short name, such as 'HOU' for the Houston Astros.
- **year** (*string* *optional*) – The requested year to pull stats from.

#### dataframe

Returns a pandas DataFrame where each row is a representation of the Game class. Rows are indexed by the boxscore string.

#### dataframe\_extended

Returns a pandas DataFrame where each row is a representation of the Boxscore class for every game in the schedule. Rows are indexed by the boxscore string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The Teams module exposes information for all MLB teams including the team name and abbreviation, the number of games they won during the season, the total number of bases they've stolen, and much more.

```
from sportsreference.mlb.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    print(team.batting_average)  # Prints the team's season batting average
```

Each Team instance contains a link to the Schedule class which enables easy iteration over all games for a particular team. A Pandas DataFrame can also be queried to easily grab all stats for all games.

```
from sportsreference.mlb.teams import Teams

teams = Teams()
for team in teams:
    schedule = team.schedule  # Returns a Schedule instance for each team
    # Returns a Pandas DataFrame of all metrics for all game Boxscores for
    # a season.
    df = team.schedule.dataframe_extended
```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```
from sportsreference.mlb.teams import Teams

for team in Teams():
    roster = team.roster  # Gets each team's roster
    for player in roster.players:
        print(player.name)  # Prints each players name on the roster
```

**class** sportsreference.mlb.teams.Team(*team\_data*, *rank*, *year=None*)

Bases: object

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as rank, name, and abbreviation, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **rank** (*int*) – A team's position in the league based on the number of points they obtained during the season.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a `string` of the team's abbreviation, such as 'HOU' for the Houston Astros.

#### at\_bats

Returns an `int` of the total number of at bats for the team.

#### average\_batter\_age

Returns a `float` of the average batter age weighted by their number of at bats plus the number of games participated in.

#### average\_pitcher\_age

Returns a `float` of the average pitcher age weighted by the number of games started, followed by the number of games played and saves.

#### away\_losses

Returns an `int` of the number of away losses during the season.

#### away\_record

Returns a `string` of the team's away record. Record is in the format 'W-L'.

#### away\_wins

Returns an `int` of the number of away wins during the season.

#### balks

Returns an `int` of the total number of times a pitcher has balked.

#### bases\_on\_balls

Returns an `int` of the number of bases on walks.

#### bases\_on\_walks\_given

Returns an `int` of the total number of bases from walks given up by a team during the season.

#### bases\_on\_walks\_given\_per\_nine\_innings

Returns a `float` of the average number of walks conceded per nine innings.

#### batters\_faced

Returns an `int` of the total number of batters all pitchers have faced during a season.

#### batting\_average

Returns a `float` of the batting average for the team. Percentage ranges from 0-1.

#### complete\_game\_shutouts

Returns an `int` of the total number of complete games where the opponent scored zero runs.

#### complete\_games

Returns an `int` of the total number of complete games a team has accumulated during the season.

#### dataframe

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'HOU'.

**doubles**

Returns an `int` of the total number of doubles hit by the team.

**earned\_runs\_against**

Returns a `float` of the average number of earned runs against for a team.

**earned\_runs\_against\_plus**

Returns an `int` of the team's average earned runs against, adjusted for the home ballpark.

**extra\_inning\_losses**

Returns an `int` of the number of losses the team has when the game has gone to extra innings.

**extra\_inning\_record**

Returns a `string` of the team's record when the game has gone to extra innings. Record is in the format 'W-L'.

**extra\_inning\_wins**

Returns an `int` of the number of wins the team has when the game has gone to extra innings.

**fielding\_independent\_pitching**

Returns a `float` of the team's effectiveness at preventing home runs, walks, batters being hit by pitches, and strikeouts.

**games**

Returns an `int` of the number of games the team has played during the season.

**games\_finished**

Returns an `int` of the number of games finished which is equivalent to the number of games played minus the number of complete games during the season.

**grounded\_into\_double\_plays**

Returns an `int` of the total number double plays grounded into by the team.

**hit\_pitcher**

Returns an `int` of the total number of times a pitcher has hit an opposing batter.

**hits**

Returns an `int` of the total number of hits during the season.

**hits\_allowed**

Returns an `int` of the total number of hits allowed during the season.

**hits\_per\_nine\_innings**

Returns a `float` of the average number of hits per nine innings by the opponent.

**home\_losses**

Returns an `int` of the number of losses at home during the season.

**home\_record**

Returns a `string` of the team's home record. Record is in the format 'W-L'.

**home\_runs**

Returns an `int` of the total number of home runs hit by the team.

**home\_runs\_against**

Returns an `int` of the total number of home runs given up during the season.

**home\_runs\_per\_nine\_innings**

Returns a `float` of the average number of home runs per nine innings by the opponent.

**home\_wins**

Returns an `int` of the number of wins at home during the season.

**innings\_pitched**

Returns a `float` of the total number of innings pitched by a team during the season.

**intentional\_bases\_on\_balls**

Returns an `int` of the total number of times a player took a base from an intentional walk.

**interleague\_record**

Returns a `string` of the team's interleague record. Record is in the format 'W-L'.

**last\_ten\_games\_record**

Returns a `string` of the team's record over the last ten games. Record is in the format 'W-L'.

**last\_thirty\_games\_record**

Returns a `string` of the team's record over the last thirty games. Record is in the format 'W-L'.

**last\_twenty\_games\_record**

Returns a `string` of the team's record over the last twenty games. Record is in the format 'W-L'.

**league**

Returns a `string` of the two letter abbreviation of the league, such as 'AL' for the American League.

**losses**

Returns an `int` of the total number of games the team lost during the season.

**losses\_last\_ten\_games**

Returns an `int` of the number of losses in the last 10 games.

**losses\_last\_thirty\_games**

Returns an `int` of the number of losses in the last 30 games.

**losses\_last\_twenty\_games**

Returns an `int` of the number of losses in the last 20 games.

**losses\_vs\_left\_handed\_pitchers**

Returns an `int` of number of losses against left-handed pitchers.

**losses\_vs\_right\_handed\_pitchers**

Returns an `int` of the number of losses against right-handed pitchers.

**losses\_vs\_teams\_over\_500**

Returns an `int` of the number of losses against teams over 500.

**losses\_vs\_teams\_under\_500**

Returns an `int` of the number of losses against teams under 500.

**luck**

Returns an `int` of the difference between the current wins and losses compared to the pythagorean wins and losses.

**name**

Returns a `string` of the team's full name, such as 'Houston Astros'.

**number\_of\_pitchers**

Returns an `int` of the total number of pitchers used during a season.

**number\_players\_used**

Returns an `int` of the number of different players used during the season.

**on\_base\_percentage**

Returns a `float` of the percentage of at bats that result in a player taking a base. Percentage ranges from 0-1.

**on\_base\_plus\_slugging\_percentage**

Returns a `float` of the sum of the on base percentage plus the slugging percentage.

**on\_base\_plus\_slugging\_percentage\_plus**

Returns an `int` of the on base percentage plus the slugging percentage, adjusted to the team's home ballpark.

**opposing\_runners\_left\_on\_base**

Returns an `int` of the total number of opponents a team has left on bases at the end of an inning.

**plate\_appearances**

Returns an `int` of the total number of plate appearances for the team.

**pythagorean\_win\_loss**

Returns a `string` of the team's expected win-loss record based on the runs scored and allowed. Record is in the format 'W-L'.

**rank**

Returns an `int` of the team's rank based on their win percentage.

**record\_vs\_left\_handed\_pitchers**

Returns a `string` of the team's record against left-handed pitchers. Record is in the format 'W-L'.

**record\_vs\_right\_handed\_pitchers**

Returns a `string` of the team's record against right-handed pitchers. Record is in the format 'W-L'.

**record\_vs\_teams\_over\_500**

Returns a `string` of the team's record against teams with a win percentage over 500. Record is in the format 'W-L'.

**record\_vs\_teams\_under\_500**

Returns a `string` of the team's record against teams with a win percentage under 500. Record is in the format 'W-L'.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**run\_difference**

Returns a `float` of the difference between the number of runs scored and the number of runs given up per game. Positive numbers indicate the team scores more per game than they are scored on.

**runners\_left\_on\_base**

Returns an `int` of the total number of runners left on base at the end of an inning.

**runs**

Returns a `float` of the average number of runs scored per game by the team.

**runs\_against**

Returns a `float` of the average number of runs scored per game by the opponent.

**runs\_allowed\_per\_game**

Returns a `float` of the average number of runs a team has allowed per game.

**runs\_batted\_in**

Returns an `int` of the total number of runs batted in by the team.

**sacrifice\_flies**

Returns an `int` of the total number of sacrifice flies the team made during the season.

**sacrifice\_hits**

Returns an `int` of the total number of sacrifice hits the team made during the season.

**saves**

Returns an `int` of the total number of saves a team has accumulated during the season.

**schedule**

Returns an instance of the Schedule class containing the team's complete schedule for the season.

**shutouts**

Returns an `int` of the total number of shutouts a team has accumulated during the season.

**simple\_rating\_system**

Returns a `float` of the average number of runs per game a team scores compared to average.

**single\_run\_losses**

Returns an `int` of the number of losses the team has when only one run is scored.

**single\_run\_record**

Returns a `string` of the team's record when only one run is scored. Record is in the format 'W-L'.

**single\_run\_wins**

Returns an `int` of the number of wins the team has when only one run is scored.

**slugging\_percentage**

Returns a `float` of the ratio of total bases gained per at bat.

**stolen\_bases**

Returns an `int` of the total number of bases stolen by the team.

**streak**

Returns a `string` of the team's current winning or losing streak, such as 'W 3' for a team on a 3-game winning streak.

**strength\_of\_schedule**

Returns a `float` denoting a team's strength of schedule, based on runs scores and conceded. Higher values result in more challenging schedules while 0.0 is an average schedule.

**strikeouts**

Returns an `int` of the total number of times a team has struck out an opponent.

**strikeouts\_per\_base\_on\_balls**

Returns a `float` of the average number of strikeouts per walk thrown by a team.

**strikeouts\_per\_nine\_innings**

Returns a `float` of the average number of strikeouts a team throws per nine innings.

**times\_caught\_stealing**

Returns an `int` of the number of times a player was caught stealing.

**times\_hit\_by\_pitch**

Returns an `int` of the total number of times a batter was hit by an opponent's pitch.

**times\_struck\_out**

Returns an `int` of the total number of times the team struck out.

**total\_bases**

Returns an `int` of the total number of bases a team has gained during the season.

**total\_runs**

Returns an `int` of the total number of runs scored during the season.

**triples**

Returns an `int` of the total number of triples hit by the team.

**whip**

Returns a `float` of the average number of walks plus hits by the opponent per inning.

**wild\_pitches**

Returns an `int` of the total number of wild pitches thrown by a team during a season.

**win\_percentage**

Returns a `float` of the number of wins divided by the number of games played during the season. Percentage ranges from 0-1.

**wins**

Returns an `int` of the total number of games the team won during the season.

**wins\_last\_ten\_games**

Returns an `int` of the number of wins in the last 10 games.

**wins\_last\_thirty\_games**

Returns an `int` of the number of wins in the last 30 games.

**wins\_last\_twenty\_games**

Returns an `int` of the number of wins in the last 20 games.

**wins\_vs\_left\_handed\_pitchers**

Returns an `int` of number of wins against left-handed pitchers.

**wins\_vs\_right\_handed\_pitchers**

Returns an `int` of the number of wins against right-handed pitchers.

**wins\_vs\_teams\_over\_500**

Returns an `int` of the number of wins against teams over 500.

**wins\_vs\_teams\_under\_500**

Returns an `int` of the number of wins against teams under 500.

**class** `sportsreference.mlb.teams.Teams` (*year=None*)

Bases: `object`

A list of all MLB teams and their stats in a given year.

Finds and retrieves a list of all MLB teams from [www.baseball-reference.com](http://www.baseball-reference.com) and creates a `Team` instance for every team that participated in the league in a given year. The `Team` class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** `year` (*string (optional)*) – The requested year to pull stats from.

**dataframes**

Returns a pandas `DataFrame` where each row is a representation of the `Team` class. Rows are indexed by the team abbreviation.

`sportsreference.mlb.teams.mlb_int_property_decorator` (*func*)

### 1.6.1.2 NBA Package

The NBA package offers multiple modules which can be use to retrieve information and statistics for the National Basketball Association, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of points scored to the number of free throws made, to the assist rate and much more. The Boxscore can be easily queried by passing a boxscore's URI on [sports-reference.com](http://sports-reference.com) which can be retrieved from the `Schedule` class (see `Schedule` module below for more information on retrieving game-specific information).



```

from sportsreference.nba.boxscore import Boxscore

game_data = Boxscore('201806080CLE')
print(game_data.away_points)  # Prints 108
print(game_data.home_points)  # Prints 85
df = game_data.dataframe  # Returns a Pandas DataFrame of game metrics

```

The Boxscore module also contains a Boxscores class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.

```

from datetime import datetime
from sportsreference.nba.boxscore import Boxscores

games_today = Boxscores(datetime.today())
print(games_today.games)  # Prints a dictionary of all matchups for today

```

The Boxscores class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```

from datetime import datetime
from sportsreference.nba.boxscore import Boxscores

# Pulls all games between and including January 1, 2018 and January 5, 2018
games = Boxscores(datetime(2018, 1, 1), datetime(2018, 1, 5))
# Prints a dictionary of all results from January 1, 2018 and January 5,
# 2018
print(games.games)

```

**class** sportsreference.nba.boxscore.Boxscore(*uri*)

Bases: object

Detailed information about the final statistics for a game.

Stores all relevant metrics for a game such as the date, time, location, result, and more advanced metrics such as the effective field goal rate, the true shooting percentage, the game's pace, and much more.

**Parameters** *uri* (*string*) – The relative link to the boxscore HTML page, such as '201710310LAL'.

**away\_assist\_percentage**

Returns a float of the percentage of the away team's field goals that were assisted. Percentage ranges from 0-100.

**away\_assists**

Returns an int of the total number of assists by the away team.

**away\_block\_percentage**

Returns a float of the percentage of 2-point field goals that were blocked by the away team. Percentage ranges from 0-100.

**away\_blocks**

Returns an int of the total number of blocks by the away team.

**away\_defensive\_rating**

Returns a float of the average number of points scored per 100 possessions by the away team.

**away\_defensive\_rebound\_percentage**

Returns a float of the percentage of available defensive rebounds the away team grabbed. Percentage

ranges from 0-100.

**away\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds by the away team.

**away\_effective\_field\_goal\_percentage**

Returns a `float` of the away team's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**away\_field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts by the away team.

**away\_field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts by the away team. Percentage ranges from 0-1.

**away\_field\_goals**

Returns an `int` of the total number of field goals made by the away team.

**away\_free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt by the away team.

**away\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts by the away team.

**away\_free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts by the away team.

**away\_free\_throws**

Returns an `int` of the total number of free throws made by the away team.

**away\_losses**

Returns an `int` of the number of games the team has lost after the conclusion of the game.

**away\_minutes\_played**

Returns an `int` of the total number of minutes the team played during the game.

**away\_offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the away team.

**away\_offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the away team grabbed. Percentage ranges from 0-100.

**away\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds by the away team.

**away\_personal\_fouls**

Returns an `int` of the total number of personal fouls by the away team.

**away\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the away team.

**away\_points**

Returns an `int` of the number of points the away team scored.

**away\_steal\_percentage**

Returns a `float` of the percentage of possessions that ended in a steal by the away team. Percentage ranges from 0-100.

**away\_steals**

Returns an `int` of the total number of steals by the away team.

**away\_three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range by the away team. Percentage ranges from 0-1.

**away\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts by the away team.

**away\_three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts by the away team. Percentage ranges from 0-1.

**away\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made by the away team.

**away\_total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the away team grabbed. Percentage ranges from 0-100.

**away\_total\_rebounds**

Returns an `int` of the total number of rebounds by the away team.

**away\_true\_shooting\_percentage**

Returns a `float` of the away team's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**away\_turnover\_percentage**

Returns a `float` of the number of times the away team turned the ball over per 100 possessions.

**away\_turnovers**

Returns an `int` of the total number of turnovers by the away team.

**away\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts by the away team.

**away\_two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts by the away team. Percentage ranges from 0-1.

**away\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made by the away team.

**away\_wins**

Returns an `int` of the number of games the team has won after the conclusion of the game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string URI that is used to instantiate the class, such as '201710310LAL'.

**date**

Returns a `string` of the date the game took place.

**home\_assist\_percentage**

Returns a `float` of the percentage of the home team's field goals that were assisted. Percentage ranges from 0-100.

**home\_assists**

Returns an `int` of the total number of assists by the home team.

**home\_block\_percentage**

Returns a `float` of the percentage of 2-point field goals that were blocked by the home team. Percentage ranges from 0-100.

**home\_blocks**

Returns an `int` of the total number of blocks by the home team.

**home\_defensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the away team.

**home\_defensive\_rebound\_percentage**

Returns a `float` of the percentage of available defensive rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds by the home team.

**home\_effective\_field\_goal\_percentage**

Returns a `float` of the home team's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**home\_field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts by the home team.

**home\_field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts by the home team. Percentage ranges from 0-1.

**home\_field\_goals**

Returns an `int` of the total number of field goals made by the home team.

**home\_free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt by the home team.

**home\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts by the home team.

**home\_free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts by the home team.

**home\_free\_throws**

Returns an `int` of the total number of free throws made by the home team.

**home\_losses**

Returns an `int` of the number of games the home team lost after the conclusion of the game.

**home\_minutes\_played**

Returns an `int` of the total number of minutes the team played during the game.

**home\_offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the home team.

**home\_offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds by the home team.

**home\_personal\_fouls**

Returns an `int` of the total number of personal fouls by the home team.

**home\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the home team.

**home\_points**

Returns an `int` of the number of points the home team scored.

**home\_steal\_percentage**

Returns a `float` of the percentage of possessions that ended in a steal by the home team. Percentage ranges from 0-100.

**home\_steals**

Returns an `int` of the total number of steals by the home team.

**home\_three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range by the home team. Percentage ranges from 0-1.

**home\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts by the home team.

**home\_three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts by the home team. Percentage ranges from 0-1.

**home\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made by the home team.

**home\_total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_total\_rebounds**

Returns an `int` of the total number of rebounds by the home team.

**home\_true\_shooting\_percentage**

Returns a `float` of the home team's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**home\_turnover\_percentage**

Returns a `float` of the number of times the home team turned the ball over per 100 possessions.

**home\_turnovers**

Returns an `int` of the total number of turnovers by the home team.

**home\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts by the home team.

**home\_two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts by the home team. Percentage ranges from 0-1.

**home\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made by the home team.

**home\_wins**

Returns an `int` of the number of games the home team won after the conclusion of the game.

**location**

Returns a `string` of the name of the venue where the game was played.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'PHO' for the Phoenix Suns.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Phoenix Suns'.

**pace**

Returns a `float` of the game's overall pace, measured by the number of possessions per 40 minutes.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'DET' for the Detroit Pistons.

**winning\_name**

Returns a `string` of the winning team's name, such as 'Detroit Pistons'.

**class** `sportsreference.nba.boxscore.BoxscorePlayer` (*player\_id*, *player\_name*,  
*player\_data*)

Bases: `sportsreference.nba.player.AbstractPlayer`

Get player stats for an individual game.

Given a player ID, such as 'hardeja01' for James Harden, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

**Parameters**

- **player\_id** (*string*) – A player's ID according to basketball-reference.com, such as 'hardeja01' for James Harden. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'LLLLLFFNN' where 'LLLLL' are the first 5 letters in the player's last name, 'FF', are the first 2 letters in the player's first name, and 'NN' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'James Harden'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values for the specified game.

**defensive\_rating**

Returns an `int` of the player's defensive rating as measured by the points allowed per 100 possessions.

**minutes\_played**

Returns a `float` of the number of game minutes the player was on the court for.

**offensive\_rating**

Returns an `int` of the player's offensive rating as measured by the points produced per 100 possessions.

**two\_point\_attempts**

Returns an `int` of the total number of two point field goals the player attempted during the season.

**two\_point\_percentage**

Returns a float of the player's two point field goal percentage during the season. Percentage ranges from 0-1.

**two\_pointers**

Returns an int of the total number of two point field goals the player made.

**class** sportsreference.nba.boxscore.Boxscores (*date*, *end\_date=None*)

Bases: object

Search for NBA games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

**Parameters**

- **date** (*datetime object*) – The date to search for any matches. The month, day, and year are required for the search, but time is not factored into the search.
- **end\_date** (*datetime object (optional)*) – Optionally specify an end date to iterate until. All boxscores starting from the date specified in the 'date' parameter up to and including the boxscores specified in the 'end\_date' parameter will be pulled. If left empty, or if 'end\_date' is prior to 'date', only the games from the day specified in the 'date' parameter will be saved.

**games**

Returns a dictionary object representing all of the games played on the requested day. Dictionary is in the following format:

```
{ 'date' : [ # 'date' is the string date in format 'MM-DD-YYYY'
    {
        'home_name': Name of the home team, such as 'Phoenix Suns'
                      (`str`),
        'home_abbr': Abbreviation for the home team, such as 'PHO'
                      (`str`),
        'away_name': Name of the away team, such as 'Houston
                      Rockets' (`str`),
        'away_abbr': Abbreviation for the away team, such as 'HOU'
                      (`str`),
        'boxscore': String representing the boxscore URI, such as
                      '201702040PHO' (`str`),
        'winning_name': Full name of the winning team, such as
                      'Houston Rockets' (`str`),
        'winning_abbr': Abbreviation for the winning team, such as
                      'HOU' (`str`),
        'losing_name': Full name of the losing team, such as
                      'Phoenix Suns' (`str`),
        'losing_abbr': Abbreviation for the losing team, such as
                      'PHO' (`str`),
        'home_score': Integer score for the home team (`int`),
        'away_score': Integer score for the away team (`int`)
    },
    { ... },
    ...
  ]
}
```

If no games were played on 'date', the list for ['date'] will be empty.

## Player

The Player module contains an abstract base class that can be inherited by both the `BoxscorePlayer` and `Player` classes in the `Boxscore` and `Roster` modules, respectively. All of the properties that appear in the `AbstractPlayer` class can be read from either of the two child classes mentioned above.

**class** `sportsreference.nba.player.AbstractPlayer` (*player\_id, player\_name, player\_data*)  
Bases: `object`

Get player information and stats for all seasons.

Given a player ID, such as 'hardeja01' for James Harden, capture all relevant stats and information like name, nationality, height/weight, career three-pointers, last season's offensive rebounds, salary, contract amount, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [basketball-reference.com](http://basketball-reference.com).

### Parameters

- **player\_id** (*string*) – A player's ID according to [basketball-reference.com](http://basketball-reference.com), such as 'hardeja01' for James Harden. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'LLLLLFFNN' where 'LLLLL' are the first 5 letters in the player's last name, 'FF', are the first 2 letters in the player's first name, and 'NN' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'James Harden'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

### **assist\_percentage**

Returns a `float` of the percentage of field goals the player assisted while on the floor. Percentage ranges from 0-100.

### **assists**

Returns an `int` of the total number of assists the player tallied during the season.

### **block\_percentage**

Returns a `float` of the percentage of opposing two-point field goal attempts that were blocked by the player while on the floor. Percentage ranges from 0-100.

### **blocks**

Returns an `int` of the total number of shots the player blocked during the season.

### **box\_plus\_minus**

Returns a `float` of the total number of points per 100 possessions the player contributed in comparison to an average player in the league.

### **defensive\_rebound\_percentage**

Returns a `float` of the percentage of available defensive rebounds the player grabbed. Percentage ranges from 0-100.

### **defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds the player grabbed during the season.



**effective\_field\_goal\_percentage**

Returns a `float` of the player's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**field\_goal\_attempts**

Returns an `int` of the total number of field goals the player attempted during the season.

**field\_goal\_percentage**

Returns a `float` of the player's field goal percentage during the season. Percentage ranges from 0-1.

**field\_goals**

Returns an `int` of the total number of field goals the player scored.

**free\_throw\_attempt\_rate**

Returns a `float` of the number of free throw attempts per field goal attempt.

**free\_throw\_attempts**

Returns an `int` of the total number of free throws the player attempted during the season.

**free\_throw\_percentage**

Returns a `float` of the player's free throw percentage during the season. Percentage ranges from 0-1.

**free\_throws**

Returns an `int` of the total number of free throws the player made during the season.

**minutes\_played**

Returns an `int` of the total number of minutes the player played.

**name**

Returns a `string` of the players name, such as 'James Harden'.

**offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the player grabbed. Percentage ranges from 0-100.

**offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds the player grabbed during the season.

**personal\_fouls**

Returns an `int` of the total number of personal fouls the player committed during the season.

**player\_id**

Returns a `string` of the player's ID on sports-reference, such as 'hardeja01' for James Harden.

**points**

Returns an `int` of the total number of points the player scored during the season.

**steal\_percentage**

Returns a `float` of the percentage of defensive possessions that ended with the player stealing the ball while on the floor. Percentage ranges from 0-100.

**steals**

Returns an `int` of the total number of steals the player tallied during the season.

**three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goals that are shot from beyond the 3-point arc. Percentage ranges from 0-1.

**three\_point\_attempts**

Returns an `int` of the total number of three point field goals the player attempted during the season.

**three\_point\_percentage**

Returns a `float` of the player's three point field goal percentage during the season. Percentage ranges from 0-1.

**three\_pointers**

Returns an `int` of the total number of three point field goals the player made.

**total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the player grabbed, both offensive and defensive. Percentage ranges from 0-100.

**total\_rebounds**

Returns an `int` of the total number of offensive and defensive rebounds the player grabbed during the season.

**true\_shooting\_percentage**

Returns a `float` of the player's true shooting percentage which takes into account two and three pointers as well as free throws. Percentage ranges from 0-1.

**turnover\_percentage**

Returns a `float` of the average number of turnovers per 100 possessions by the player.

**turnovers**

Returns an `int` of the total number of times the player turned the ball over during the season for any reason.

**usage\_percentage**

Returns a `float` of the percentage of plays the player is involved in while on the floor. Percentage ranges from 0-100.

## Roster

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the `Player` class which has detailed information ranging from career points totals to single-season stats and player height, weight, and nationality. The following is an example on collecting career information for James Harden:

```
from sportsreference.nba.roster import Player

james_harden = Player('hardeja01')
print(james_harden.name) # Prints 'James Harden'
print(james_harden.points) # Prints Harden's career points total
# Prints a Pandas DataFrame of all relevant Harden stats per season
print(james_harden.dataframe)
print(james_harden.salary) # Prints Harden's career earnings
print(james_harden.contract) # Prints Harden's contract by yearly wages
```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific season, call the class instance with the season string. All future property requests will return the season-specific stats.

```
from sportsreference.nba.roster import Player

james_harden = Player('hardeja01') # Currently pulling career stats
print(james_harden.points) # Prints Harden's CAREER points total
# Prints Harden's points total only for the 2017-18 season.
print(james_harden('2017-18').points)
# Prints the number of games Harden played in the 2017-18 season.
print(james_harden.games_played)
```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.nba.roster import Player

james_harden = Player('hardeja01') # Currently pulling career stats
# Prints Harden's points total only for the 2017-18 season.
print(james_harden('2017-18').points)
print(james_harden('Career').points) # Prints Harden's career points total
```

In addition, the Roster module also contains the Roster class which can be used to pull all players on a team's roster during a given season and creates instances of the Player class for each team member and adds them to a list to be easily queried.

```
from sportsreference.nba.roster import Roster

houston = Roster('HOU')
for player in houston.players:
    # Prints the name of all players who played for Houston in the most
    # recent season.
    print(player.name)
```

**class** sportsreference.nba.roster.**Player** (*player\_id*)  
 Bases: *sportsreference.nba.player.AbstractPlayer*

Get player information and stats for all seasons.

Given a player ID, such as 'hardeja01' for James Harden, capture all relevant stats and information like name, nationality, height/weight, career three-pointers, last season's offensive rebounds, salary, contract amount, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [basketball-reference.com](http://basketball-reference.com).

**Parameters** *player\_id* (*string*) – A player's ID according to [basketball-reference.com](http://basketball-reference.com), such as 'hardeja01' for James Harden. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'LLLLLFFNN' where 'LLLLL' are the first 5 letters in the player's last name, 'FF', are the first 2 letters in the player's first name, and 'NN' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.

#### **and\_ones**

Returns an *int* of the total number of times the player was fouled in the act of shooting and made the basket.

#### **birth\_date**

Returns a *datetime* object of the day and year the player was born.

#### **blocking\_fouls**

Returns an *int* of the total number of blocking fouls the player committed.

#### **center\_percentage**

Returns an *int* of the percentage of time the player spent as a center. Percentage ranges from 0-100 and is rounded to the nearest whole number.

#### **contract**

Returns a *dictionary* of the player's contract details where the key is a *string* of the season, such as '2018-19', and the value is a *string* of the salary, such as '\$40,000,000'.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values where each index is a different season plus the career stats.

**defensive\_box\_plus\_minus**

Returns a `float` of the number of defensive points per 100 possessions the player contributed in comparison to an average player in the league.

**defensive\_win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his defensive plays.

**dunks**

Returns an `int` of the total number of dunks the player made during the season.

**field\_goal\_perc\_sixteen\_foot\_plus\_two\_pointers**

Returns a `float` of the player's field goal percentage for shots that are greater than sixteen feet from the basket, but in front of or on the three point arc. Percentage ranges from 0-1.

**field\_goal\_perc\_ten\_to\_sixteen\_feet**

Returns a `float` of the player's field goal percentage for shots between ten and sixteen feet from the basket. Percentage ranges from 0-1.

**field\_goal\_perc\_three\_to\_ten\_feet**

Returns a `float` of the player's field goal percentage for shots between three and ten feet from the basket. Percentage ranges from 0-1.

**field\_goal\_perc\_zero\_to\_three\_feet**

Returns a `float` of the player's field goal percentage for shots between zero and three feet from the basket. Percentage ranges from 0-1.

**games\_played**

Returns an `int` of the number of games the player participated in.

**games\_started**

Returns an `int` of the number of games the player started.

**half\_court\_heaves**

Returns an `int` of the number of shots the player took from beyond mid-court.

**half\_court\_heaves\_made**

Returns an `int` of the number of shots the player made from beyond mid-court.

**height**

Returns a `string` of the player's height in the format "feet-inches".

**lost\_ball\_turnovers**

Returns an `int` of the total number of turnovers the player committed due to losing the ball.

**nationality**

Returns a `string` constant denoting which country the player originates from.

**net\_plus\_minus**

Returns a `float` of the net number of points the player contributes to the team per 100 possessions regardless of being on the floor or not.

**offensive\_box\_plus\_minus**

Returns a `float` of the number of offensive points per 100 possessions the player contributed in comparison to an average player in the league.

**offensive\_fouls**

Returns an `int` of the total number of offensive fouls the player committed.

**offensive\_win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his offensive plays.

**on\_court\_plus\_minus**

Returns a `float` of the number of points the player contributes to the team while on the court per 100 possessions.

**other\_turnovers**

Returns an `int` of the total number of all other non-passing/ dribbling turnovers the player committed.

**passing\_turnovers**

Returns an `int` of the total number of turnovers the player committed due to a bad pass.

**percentage\_field\_goals\_as\_dunks**

Returns a `float` of the percentage of the player's shot attempts that are dunks. Percentage ranges from 0-1.

**percentage\_of\_three\_pointers\_from\_corner**

Returns a `float` of the percentage of 3-point shots the player attempted from the corner. Percentage ranges from 0-1.

**percentage\_shots\_three\_pointers**

Returns a `float` of the percentage of shots the player takes from beyond the three point arc. Percentage ranges from 0-1.

**percentage\_shots\_two\_pointers**

Returns a `float` of the percentage of shots the player takes that are 2-pointers. Percentage ranges from 0-1.

**percentage\_sixteen\_foot\_plus\_two\_pointers**

Returns a `float` of the percentage of shots the player takes that are greater than sixteen feet from the basket, but in front of or on the three point arc. Percentage ranges from 0-1.

**percentage\_ten\_to\_sixteen\_footers**

Returns a `float` of the percentage of shots the player takes from ten to sixteen feet from the basket. Percentage ranges from 0-1.

**percentage\_three\_to\_ten\_footers**

Returns a `float` of the percentage of shots the player takes from three to ten feet from the basket. Percentage ranges from 0-1.

**percentage\_zero\_to\_three\_footers**

Returns a `float` of the percentage of shots the player takes from zero to three feet from the basket. Percentage ranges from 0-1.

**player\_efficiency\_rating**

Returns a `float` of the player's efficiency rating which represents the player's relative production level. An average player in the league has an efficiency rating of 15.

**point\_guard\_percentage**

Returns an `int` of the percentage of time the player spent as a point guard. Percentage ranges from 0-100 and is rounded to the nearest whole number.

**points\_generated\_by\_assists**

Returns an `int` of the total number of points the player generated as a result of him assisting the shooter.

**position**

Returns a `string` constant of the player's primary position.

**power\_forward\_percentage**

Returns an `int` of the percentage of time the player spent as a power forward. Percentage ranges from 0-100 and is rounded to the nearest whole number.

**salary**

Returns an `int` of the player's annual salary rounded down.

**season**

Returns a `string` of the season in the format 'YYYY-YY', such as '2017-18'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**shooting\_distance**

Returns a `float` of the average distance the player takes a shot from in feet.

**shooting\_fouls**

Returns an `int` of the total number of shooting fouls the player committed.

**shooting\_fouls\_drawn**

Returns an `int` of the total number of shooting fouls the player drew during the season.

**shooting\_guard\_percentage**

Returns an `int` of the percentage of time the player spent as a shooting guard. Percentage ranges from 0-100 and is rounded to the nearest whole number.

**shots\_blocked**

Returns an `int` of the total number of shots the player took that were blocked by an opposing player.

**small\_forward\_percentage**

Returns an `int` of the percentage of time the player spent as a small forward. Percentage ranges from 0-100 and is rounded to the nearest whole number.

**take\_fouls**

Returns an `int` of the total number of take fouls the player committed by taking a foul before the offensive player has a chance to make a shooting motion.

**team\_abbreviation**

Returns a `string` of the abbreviation for the team the player plays for, such as 'HOU' for James Harden.

**three\_point\_shot\_percentage\_from\_corner**

Returns a `float` of the percentage of 3-pointers from the corner that went in. Percentage ranges from 0-1.

**three\_pointers\_assisted\_percentage**

Returns a `float` of the percentage of 3-point field goals by the player that are assisted. Percentage ranges from 0-1.

**two\_point\_attempts**

Returns an `int` of the total number of two point field goals the player attempted during the season.

**two\_point\_percentage**

Returns a `float` of the player's two point field goal percentage during the season. Percentage ranges from 0-1.

**two\_pointers**

Returns an `int` of the total number of two point field goals the player made.

**two\_pointers\_assisted\_percentage**

Returns a `float` of the percentage of 2-point field goals by the player that are assisted. Percentage ranges from 0-1.

**value\_over\_replacement\_player**

Returns a `float` of the total number of points per 100 team possessions the player contributed compared

to a replacement-level player (who has an average score of -2.0). This value is prorated for an 82-game season.

#### **weight**

Returns an `int` of the player's weight in pounds.

#### **win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his offensive and defensive plays.

#### **win\_shares\_per\_48\_minutes**

Returns a `float` of the number of wins the player contributed to the team per 48 minutes of playtime. An average player has a contribution of 0.100.

**class** `sportsreference.nba.roster.Roster` (*team*, *year=None*, *slim=False*)

Bases: `object`

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the `Player` class for each player, containing a detailed list of the players statistics and information.

#### **Parameters**

- **team** (*string*) – The team's 3-letter abbreviation, such as 'HOU' for the Houston Rockets.
- **year** (*string (optional)*) – The 4-digit year to pull the roster from, such as '2018'. If left blank, defaults to the most recent season.
- **slim** (*boolean (optional)*) – Set to `True` to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to `False`.

#### **players**

Returns a `list` of player instances for each player on the requested team's roster if the `slim` property is `False` when calling the `Roster` class. If the `slim` property is `True`, returns a `dictionary` where each key is a string of the player's ID and each value is the player's first and last name as listed on the roster page.

## Schedule

The `Schedule` module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.nba.schedule import Schedule

houston_schedule = Schedule('HOU')
for game in houston_schedule:
    print(game.date)    # Prints the date the game was played
    print(game.result)  # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** `sportsreference.nba.schedule.Game` (*game\_data*, *playoffs=False*)

Bases: `object`

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

**Parameters** `game_data` (*string*) – The row containing the specified game information.

**boxscore**

Returns an instance of the Boxscore class containing more detailed stats on the game.

**boxscore\_index**

Returns a *string* of the URI for a boxscore which can be used to access or index a game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame representing the Boxscore class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property. The index for the DataFrame is the boxscore string.

**date**

Returns a *string* of the date the game took place at, such as 'Wed, Oct 18, 2017'.

**datetime**

Returns a datetime object to indicate the month, day, and year the game took place.

**game**

Returns an *int* to indicate which game in the season was requested. The first game of the season returns 1.

**location**

Returns a *string* constant to indicate whether the game was played in the team's home arena or on the road.

**losses**

Returns an *int* of the number of losses the team has in the season after the completion of the listed game.

**opponent\_abbr**

Returns a *string* of the opponent's 3-letter abbreviation, such as 'CHI' for the Chicago Bulls.

**opponent\_name**

Returns a *string* of the opponent's name, such as 'Chicago Bulls'.

**playoffs**

Returns a *boolean* variable which evaluates to True when the game was played in the playoffs and returns False if the game took place in the regular season.

**points\_allowed**

Returns an *int* of the number of points the team allowed during the game.

**points\_scored**

Returns an *int* of the number of points the team scored during the game.

**result**

Returns a *string* constant to indicate whether the team won or lost the game.

**streak**

Returns a *string* of the team's current streak after the conclusion of the listed game, such as 'W 3' for a 3-game winning streak.

**time**

Returns a *string* of the time the game started in Eastern Time, such as '8:01p'.



**wins**

Returns an int of the number of wins the team has in the season after the completion of the listed game.

**class** sportsreference.nba.schedule.**Schedule** (*abbreviation*, *year=None*)

Bases: object

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

**Parameters**

- **abbreviation** (*string*) – A team's short name, such as 'PHO' for the Phoenix Suns.
- **year** (*string* (*optional*)) – The requested year to pull stats from.

**dataframe**

Returns a pandas DataFrame where each row is a representation of the Game class. Rows are indexed by the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame where each row is a representation of the Boxscore class for every game in the schedule. Rows are indexed by the boxscore string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The Teams module exposes information for all NBA teams including the team name and abbreviation, the number of games they won during the season, the total number of shots they've blocked, and much more.

```
from sportsreference.nba.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    print(team.blocks)  # Prints the team's total blocked shots
```

Each Team instance contains a link to the Schedule class which enables easy iteration over all games for a particular team. A Pandas DataFrame can also be queried to easily grab all stats for all games.

```
from sportsreference.nba.teams import Teams

teams = Teams()
for team in teams:
    schedule = team.schedule  # Returns a Schedule instance for each team
    # Returns a Pandas DataFrame of all metrics for all game Boxscores for
    # a season.
    df = team.schedule.dataframe_extended
```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```
from sportsreference.nba.teams import Teams

teams = Teams()
for team in teams:
    # Creates an instance of the roster class for each player on the team.
    roster = team.roster
```

(continues on next page)

(continued from previous page)

```
for player in roster.players:
    print(player.name) # Prints the name of each player on the team.
```

**class** sportsreference.nba.teams.**Team**(team\_data, rank, year=None)

Bases: object

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as rank, name, and abbreviation, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **rank** (*int*) – A team's position in the league based on the number of points they obtained during the season.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a *string* of the team's abbreviation, such as 'DET' for the Detroit Pistons.

#### assists

Returns an *int* of the total number of field goals that were assisted.

#### blocks

Returns an *int* of the total number of times the team blocked an opponent's shot.

#### dataframe

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'DET'.

#### defensive\_rebounds

Returns an *int* of the total number of defensive rebounds the team has grabbed.

#### field\_goal\_attempts

Returns an *int* of the total number of field goals the team has attempted during the season.

#### field\_goal\_percentage

Returns a *float* of the percentage of field goals made divided by the number of attempts. Percentage ranges from 0-1.

#### field\_goals

Returns an *int* of the total number of field goals the team has made during the season.

#### free\_throw\_attempts

Returns an *int* of the total number of free throw attempts during the season.

#### free\_throw\_percentage

Returns a *float* of the percentage of free throws made divided by the attempts. Percentage ranges from 0-1.

#### free\_throws

Returns an *int* of the total number of free throws made during the season.

#### games\_played

Returns an *int* of the total number of games the team has played during the season.

**minutes\_played**

Returns an `int` of the total number of minutes played by all players on the team during the season.

**name**

Returns a `string` of the team's full name, such as 'Detroit Pistons'.

**offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds the team has grabbed.

**opp\_assists**

Returns an `int` of the total number of field goals that were assisted by the opponent.

**opp\_blocks**

Returns an `int` of the total number of times the opponent blocked the team's shot.

**opp\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds the opponent grabbed.

**opp\_field\_goal\_attempts**

Returns an `int` of the total number of field goals the opponents attempted during the season.

**opp\_field\_goal\_percentage**

Returns a `float` of the percentage of field goals made divided by the number of attempts by the opponent. Percentage ranges from 0-1.

**opp\_field\_goals**

Returns an `int` of the total number of field goals the opponents made during the season.

**opp\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts during the season by the opponent.

**opp\_free\_throw\_percentage**

Returns a `float` of the percentage of free throws made divided by the attempts by the opponent. Percentage ranges from 0-1.

**opp\_free\_throws**

Returns an `int` of the total number of free throws made during the season by the opponent.

**opp\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds the opponent grabbed.

**opp\_personal\_fouls**

Returns an `int` of the total number of times the opponent fouled the team.

**opp\_points**

Returns an `int` of the total number of points the team has been scored on during the season.

**opp\_steals**

Returns an `int` of the total number of times the opponent stole the ball from the team.

**opp\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goals the opponent attempted during the season.

**opp\_three\_point\_field\_goal\_percentage**

Returns a `float` of the percentage of three point field goals made divided by the number of attempts by the opponent. Percentage ranges from 0-1.

**opp\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals the opponent made during the season.

**opp\_total\_rebounds**

Returns an `int` of the total number of rebounds the opponent grabbed.

**opp\_turnovers**

Returns an `int` of the total number of times the opponent turned the ball over.

**opp\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goals the opponent attempted during the season.

**opp\_two\_point\_field\_goal\_percentage**

Returns a `float` of the percentage of two point field goals made divided by the number of attempts by the opponent. Percentage ranges from 0-1.

**opp\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals the opponent made during the season.

**personal\_fouls**

Returns an `int` of the total number of times the team has fouled an opponent.

**points**

Returns an `int` of the total number of points the team has scored during the season.

**rank**

Returns an `int` of the team's rank based on the number of points they score per game.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**schedule**

Returns an instance of the Schedule class containing the team's complete schedule for the season.

**steals**

Returns an `int` of the total number of times the team stole the ball from the opponent.

**three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goals the team has attempted during the season.

**three\_point\_field\_goal\_percentage**

Returns a `float` of the percentage of three point field goals made divided by the number of attempts. Percentage ranges from 0-1.

**three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals the team has made during the season.

**total\_rebounds**

Returns an `int` of the total number of rebounds the team has grabbed.

**turnovers**

Returns an `int` of the total number of times the team has turned the ball over.

**two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goals the team has attempted during the season.

**two\_point\_field\_goal\_percentage**

Returns a `float` of the percentage of two point field goals made divided by the number of attempts. Percentage ranges from 0-1.

**two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals the team has made during the season.

**class** sportsreference.nba.teams.Teams (*year=None*)

Bases: `object`

A list of all NBA teams and their stats in a given year.

Finds and retrieves a list of all NBA teams from [www.basketball-reference.com](http://www.basketball-reference.com) and creates a Team instance for every team that participated in the league in a given year. The Team class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** `year` (*string (optional)*) – The requested year to pull stats from.

**dataframes**

Returns a pandas DataFrame where each row is a representation of the Team class. Rows are indexed by the team abbreviation.

### 1.6.1.3 NCAAB Package

The NCAAB package offers multiple modules which can be used to retrieve information and statistics for Men's Division I College Basketball, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of points scored to the number of blocked shots, to the assist percentage and much more. The Boxscore can be easily queried by passing a boxscore's URI on [sports-reference.com](http://sports-reference.com) which can be retrieved from the Schedule class (see Schedule module below for more information on retrieving game-specific information).

```
from sportsreference.ncaab.boxscore import Boxscore

game_data = Boxscore('2018-04-02-21-villanova')
print(game_data.home_points)  # Prints 79
print(game_data.away_points)  # Prints 62
df = game_data.dataframe  # Returns a Pandas DataFrame of game metrics
```

The Boxscore module also contains a Boxscores class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.

```
from datetime import datetime
from sportsreference.ncaab.boxscore import Boxscores

games_today = Boxscores(datetime.today())
print(games_today.games)  # Prints a dictionary of all matchups for today
```

The Boxscores class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```
from datetime import datetime
from sportsreference.ncaab.boxscore import Boxscores

# Pulls all games between and including November 11, 2017 and November 12,
# 2017
games = Boxscores(datetime(2017, 11, 11), datetime(2017, 11, 12))
# Prints a dictionary of all results from November 11, 2017 and November 12,
# 2017
print(games.games)
```

```
class sportsreference.ncaab.boxscore.Boxscore(uri)
    Bases: object
```

Detailed information about the final statistics for a game.

Stores all relevant metrics for a game such as the date, time, location, result, and more advanced metrics such as the effective field goal rate, the true shooting percentage, the game's pace, and much more.

**Parameters** `uri` (*string*) – The relative link to the boxscore HTML page, such as '2017-11-10-21-kansas'.

**away\_assist\_percentage**

Returns a `float` of the percentage of the away team's field goals that were assisted. Percentage ranges from 0-100.

**away\_assists**

Returns an `int` of the total number of assists by the away team.

**away\_block\_percentage**

Returns a `float` of the percentage of 2-point field goals that were blocked by the away team. Percentage ranges from 0-100.

**away\_blocks**

Returns an `int` of the total number of blocks by the away team.

**away\_defensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the away team.

**away\_defensive\_rebound\_percentage**

Returns a `float` of the percentage of available defensive rebounds the away team grabbed. Percentage ranges from 0-100.

**away\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds by the away team.

**away\_effective\_field\_goal\_percentage**

Returns a `float` of the away team's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**away\_field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts by the away team.

**away\_field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts by the away team. Percentage ranges from 0-1.

**away\_field\_goals**

Returns an `int` of the total number of field goals made by the away team.

**away\_free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt by the away team.

**away\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts by the away team.

**away\_free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts by the away team.

**away\_free\_throws**

Returns an `int` of the total number of free throws made by the away team.

**away\_losses**

Returns an `int` of the number of games the team has lost after the conclusion of the game.

**away\_minutes\_played**

Returns an `int` of the total number of minutes the team played during the game.

**away\_offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the away team.

**away\_offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the away team grabbed. Percentage ranges from 0-100.

**away\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds by the away team.

**away\_personal\_fouls**

Returns an `int` of the total number of personal fouls by the away team.

**away\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the away team.

**away\_points**

Returns an `int` of the number of points the away team scored.

**away\_ranking**

Returns an `int` of the away team's ranking during the week, or `None` if the team wasn't ranked.

**away\_steal\_percentage**

Returns a `float` of the percentage of possessions that ended in a steal by the away team. Percentage ranges from 0-100.

**away\_steals**

Returns an `int` of the total number of steals by the away team.

**away\_three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range by the away team. Percentage ranges from 0-1.

**away\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts by the away team.

**away\_three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts by the away team. Percentage ranges from 0-1.

**away\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made by the away team.

**away\_total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the away team grabbed. Percentage ranges from 0-100.

**away\_total\_rebounds**

Returns an `int` of the total number of rebounds by the away team.

**away\_true\_shooting\_percentage**

Returns a `float` of the away team's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**away\_turnover\_percentage**

Returns a `float` of the number of times the away team turned the ball over per 100 possessions.

**away\_turnovers**

Returns an `int` of the total number of turnovers by the away team.

**away\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts by the away team.

**away\_two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts by the away team. Percentage ranges from 0-1.

**away\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made by the away team.

**away\_win\_percentage**

Returns a `float` of the percentage of games the away team has won after the conclusion of the game. Percentage ranges from 0-1.

**away\_wins**

Returns an `int` of the number of games the team has won after the conclusion of the game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string URI that is used to instantiate the class, such as '2017-11-10-21-kansas'.

**date**

Returns a `string` of the date the game took place.

**home\_assist\_percentage**

Returns a `float` of the percentage of the home team's field goals that were assisted. Percentage ranges from 0-100.

**home\_assists**

Returns an `int` of the total number of assists by the home team.

**home\_block\_percentage**

Returns a `float` of the percentage of 2-point field goals that were blocked by the home team. Percentage ranges from 0-100.

**home\_blocks**

Returns an `int` of the total number of blocks by the home team.

**home\_defensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the away team.

**home\_defensive\_rebound\_percentage**

Returns a `float` of the percentage of available defensive rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds by the home team.

**home\_effective\_field\_goal\_percentage**

Returns a `float` of the home team's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**home\_field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts by the home team.

**home\_field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts by the home team. Percentage ranges from 0-1.

**home\_field\_goals**

Returns an `int` of the total number of field goals made by the home team.



**home\_free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt by the home team.

**home\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts by the home team.

**home\_free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts by the home team.

**home\_free\_throws**

Returns an `int` of the total number of free throws made by the home team.

**home\_losses**

Returns an `int` of the number of games the home team lost after the conclusion of the game.

**home\_minutes\_played**

Returns an `int` of the total number of minutes the team played during the game.

**home\_offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the home team.

**home\_offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds by the home team.

**home\_personal\_fouls**

Returns an `int` of the total number of personal fouls by the home team.

**home\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the home team.

**home\_points**

Returns an `int` of the number of points the home team scored.

**home\_ranking**

Returns an `int` of the home team's ranking during the week, or `None` if they were not ranked.

**home\_steal\_percentage**

Returns a `float` of the percentage of possessions that ended in a steal by the home team. Percentage ranges from 0-100.

**home\_steals**

Returns an `int` of the total number of steals by the home team.

**home\_three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range by the home team. Percentage ranges from 0-1.

**home\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts by the home team.

**home\_three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts by the home team. Percentage ranges from 0-1.

**home\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made by the home team.

**home\_total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the home team grabbed. Percentage ranges from 0-100.

**home\_total\_rebounds**

Returns an `int` of the total number of rebounds by the home team.

**home\_true\_shooting\_percentage**

Returns a `float` of the home team's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**home\_turnover\_percentage**

Returns a `float` of the number of times the home team turned the ball over per 100 possessions.

**home\_turnovers**

Returns an `int` of the total number of turnovers by the home team.

**home\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts by the home team.

**home\_two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts by the home team. Percentage ranges from 0-1.

**home\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made by the home team.

**home\_win\_percentage**

Returns a `float` of the percentage of games the home team has won after the conclusion of the game. Percentage ranges from 0-1.

**home\_wins**

Returns an `int` of the number of games the home team won after the conclusion of the game.

**location**

Returns a `string` of the name of the venue where the game was played.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'INDIANA' for the Indiana Hoosiers.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Indiana' Hoosiers'.

**pace**

Returns a `float` of the game's overall pace, measured by the number of possessions per 40 minutes.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'PURDUE' for the Purdue Boilermakers.

**winning\_name**

Returns a `string` of the winning team's name, such as 'Purdue Boilermakers'.

**class** `sportsreference.ncaab.boxscore.BoxscorePlayer` (*player\_id*, *player\_name*,  
*player\_data*)

Bases: `sportsreference.ncaab.player.AbstractPlayer`

Get player stats for an individual game.

Given a player ID, such as 'carsen-edwards-1' for Carsen Edwards, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

#### Parameters

- **player\_id**(*string*) – A player’s ID accorsing to sports-reference.com, such as ‘carsen-edwards-1’ for Carsen Edwards. The player ID can be found by navigating to the player’s stats page and getting the string between the final slash and the ‘.html’ in the URL. In general, the ID is in the format ‘first-last-N’ where ‘first’ is the player’s first name in lowercase, ‘last’ is the player’s last name in lowercase, and ‘N’ is a number starting at ‘1’ for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name**(*string*) – A string representing the player’s first and last name, such as ‘Carsen Edwards’.
- **player\_data**(*string*) – A string representation of the player’s HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

#### dataframe

Returns a `pandas DataFrame` containing all other relevant class properties and values for the specified game.

#### defensive\_rating

Returns an `int` of the player’s defensive rating as measured by the points allowed per 100 possessions.

#### offensive\_rating

Returns an `int` of the player’s offensive rating as measured by the points produced per 100 possessions.

**class** `sportsreference.ncaab.boxscore.Boxscores` (*date*, *end\_date=None*)

Bases: `object`

Search for NCAAB games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, a boolean value which indicates if the game is between two Division-I teams or not, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

#### Parameters

- **date** (*datetime object*) – The date to search for any matches. The month, day, and year are required for the search, but time is not factored into the search.
- **end\_date** (*datetime object*) – Optionally specify an end date to iterate until. All boxscores starting from the date specified in the ‘date’ parameter up to and including the boxscores specified in the ‘end\_date’ parameter will be pulled. If left empty, or if ‘end\_date’ is prior to ‘date’, only the games from the day specified in the ‘date’ parameter will be saved.

#### games

Returns a `dictionary object` representing all of the games played on the requested day. Dictionary is in the following format:

```
{ 'date' : [ # 'date' is the string date in format 'MM-DD-YYYY'
  {
    'home_name': Name of the home team, such as 'Purdue
                  Boilermakers' (`str`),
    'home_abbr': Abbreviation for the home team, such as
```

(continues on next page)

(continued from previous page)

```

        'PURDUE' (`str`),
    'away_name': Name of the away team, such as 'Indiana
                  Hoosiers' (`str`),
    'away_abbr': Abbreviation for the away team, such as
                  'INDIANA' (`str`),
    'boxscore': String representing the boxscore URI, such as
                  '2018-01-28-15-indiana' (`str`),
    'non_di': Boolean value which evaluates to True when at
               least one of the teams does not compete in NCAA
               Division-I basketball (`bool`),
    'top_25': Boolean value which evaluates to True when at
              least one of the teams is ranked in the AP Top 25
              polls (`bool`),
    'winning_name': Full name of the winning team, such as
                    'Purdue Boilermakers' (`str`),
    'winning_abbr': Abbreviation for the winning team, such as
                    'PURDUE' (`str`),
    'losing_name': Full name of the losing team, such as
                   'Indiana Hoosiers' (`str`),
    'losing_abbr': Abbreviation for the losing team, such as
                   'INDIANA' (`str`),
    'home_score': Integer score for the home team (`int`),
    'home_rank': Integer representing the home team's rank
                 (`int`),
    'away_score': Integer score for the away team (`int`),
    'away_rank': Integer representing the away team's rank
                 (`int`)
    },
    { ... },
    ...
]
}

```

If no games were played on ‘date’, the list for [‘date’] will be empty.

## Conferences

The Conference module allows conferences to be pulled for any season using the Conferences class. Accessing the class properties exposes various dictionaries containing the team and conference abbreviations as well as other information. To get a list of conference abbreviations for each team, query the `team_conference` property.

```

from sportsreference.ncaab.conferences import Conferences

conferences = Conferences()
# Prints a dictionary of the team abbreviation as a key and conference
# abbreviation as the value.
print(conferences.team_conference)

```

The conferences property can also be queried to provide more details on the teams in every conference.

```

from sportsreference.ncaab.conferences import Conferences

conferences = Conferences()
# Prints a dictionary where each key is the conference abbreviation and
# each value is a dictionary containing the full conference name as well as

```

(continues on next page)

(continued from previous page)

```
# another dictionary of all teams in the conference, including name and
# abbreviation for each team.
print(conferences.conferences)
```

```
class sportsreference.ncaab.conferences.Conference (conference_abbreviation,
                                                    year=None)
```

Bases: object

Find teams that participated in a particular conference.

Create a dictionary which includes the names and abbreviations for all teams that participated in a conference during a given year.

#### Parameters

- **conference\_abbreviation** (*string*) – A string of the requested conference’s abbreviation, such as ‘big-12’.
- **year** (*string (optional)*) – A string of the requested year to pull conference information from. Defaults to the most recent season.

#### teams

Returns a dictionary of team names and abbreviations where each key is a string of the team abbreviation and each value is a string of the full team name.

```
class sportsreference.ncaab.conferences.Conferences (year=None)
```

Bases: object

Get all conferences and teams for a season.

Retrieve a list of all conferences and teams that participated in the conference for each team in the season. The included properties allow flexibility in queries to either get the conference abbreviation for a given team, or get more detailed information including all teams for each conference.

**Parameters** **year** (*string (optional)*) – A string of the requested year to pull conferences from. Defaults to the most recent season.

#### conferences

Returns a dictionary of conference names and abbreviations where each key is a string of the abbreviation and each value is a dictionary containing the full conference name and another dictionary with individual team information. The overall dictionary is in the following structure:

```
{
    abbreviation, ie 'big-12' (str): {
        'name': Full conference name, such as 'Big 12 Conference'
            (str),
        'teams': {
            team abbreviation, such as 'kansas' (str): Full team
                name, such as 'Kansas' (str),
            ...
        }
    },
    ...
}
```

#### team\_conference

Returns a dictionary of conference abbreviations for each team where each key is a string of the team abbreviation and each value is a string of the conference abbreviation.

## Player

The Player module contains an abstract base class that can be inherited by both the `BoxscorePlayer` and `Player` classes in the `Boxscore` and `Roster` modules, respectively. All of the properties that appear in the `AbstractPlayer` class can be read from either of the two child classes mentioned above.

```
class sportsreference.ncaab.player.AbstractPlayer(player_id, player_name,  
                                                player_data)
```

Bases: `object`

Get player information and stats for all seasons.

Given a player ID, such as 'carsen-edwards-1' for Carsen Edwards, capture all relevant stats and information like name, height/weight, career three-pointers, last season's offensive rebounds, offensive points contributed, and much more.

### Parameters

- **player\_id** (*string*) – A player's ID according to sports-reference.com, such as 'carsen-edwards-1' for Carsen Edwards. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'first-last-N' where 'first' is the player's first name in lowercase, 'last' is the player's last name in lowercase, and 'N' is a number starting at '1' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'Carsen Edwards'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

### **assist\_percentage**

Returns a `float` of the percentage of field goals the player assisted while on the floor. Percentage ranges from 0-100.

### **assists**

Returns an `int` of the total number of assists the player tallied during the season.

### **block\_percentage**

Returns a `float` of the percentage of opposing two-point field goal attempts that were blocked by the player while on the floor. Percentage ranges from 0-100.

### **blocks**

Returns an `int` of the total number of shots the player blocked during the season.

### **defensive\_rebound\_percentage**

Returns a `float` of the percentage of available defensive rebounds the player grabbed. Percentage ranges from 0-100.

### **defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds the player grabbed during the season.

### **effective\_field\_goal\_percentage**

Returns a `float` of the player's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

### **field\_goal\_attempts**

Returns an `int` of the total number of field goals the player attempted during the season.

### **field\_goal\_percentage**

Returns a `float` of the player's field goal percentage during the season. Percentage ranges from 0-1.

**field\_goals**

Returns an `int` of the total number of field goals the player scored.

**free\_throw\_attempt\_rate**

Returns a `float` of the number of free throw attempts per field goal attempt.

**free\_throw\_attempts**

Returns an `int` of the total number of free throws the player attempted during the season.

**free\_throw\_percentage**

Returns a `float` of the player's free throw percentage during the season. Percentage ranges from 0-1.

**free\_throws**

Returns an `int` of the total number of free throws the player made during the season.

**minutes\_played**

Returns an `int` of the total number of minutes the player played.

**name**

Returns a `string` of the players name, such as 'Carsen Edwards'.

**offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the player grabbed. Percentage ranges from 0-100.

**offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds the player grabbed during the season.

**personal\_fouls**

Returns an `int` of the total number of personal fouls the player committed during the season.

**player\_id**

Returns a `string` of the player's ID on sports-reference, such as 'carsen-edwards-1' for Carsen Edwards.

**points**

Returns an `int` of the total number of points the player scored during the season.

**steal\_percentage**

Returns a `float` of the percentage of defensive possessions that ended with the player stealing the ball while on the floor. Percentage ranges from 0-100.

**steals**

Returns an `int` of the total number of steals the player tallied during the season.

**three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goals that are shot from beyond the 3-point arc. Percentage ranges from 0-1.

**three\_point\_attempts**

Returns an `int` of the total number of three point field goals the player attempted during the season.

**three\_point\_percentage**

Returns a `float` of the player's three point field goal percentage during the season. Percentage ranges from 0-1.

**three\_pointers**

Returns an `int` of the total number of three point field goals the player made.

**total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the player grabbed, both offensive and defensive. Percentage ranges from 0-100.

**total\_rebounds**

Returns an `int` of the total number of offensive and defensive rebounds the player grabbed during the season.

**true\_shooting\_percentage**

Returns a `float` of the player's true shooting percentage which takes into account two and three pointers as well as free throws. Percentage ranges from 0-1.

**turnover\_percentage**

Returns a `float` of the average number of turnovers per 100 possessions by the player.

**turnovers**

Returns an `int` of the total number of times the player turned the ball over during the season for any reason.

**two\_point\_attempts**

Returns an `int` of the total number of two point field goals the player attempted during the season.

**two\_point\_percentage**

Returns a `float` of the player's two point field goal percentage during the season. Percentage ranges from 0-1.

**two\_pointers**

Returns an `int` of the total number of two point field goals the player made.

**usage\_percentage**

Returns a `float` of the percentage of plays the player is involved in while on the floor. Percentage ranges from 0-100.

## Rankings

The Rankings module includes the `Rankings` class which can be used to easily query the NCAA Men's Division-I Basketball rankings published by the Associated Press on a week-by-week basis. Different formats can be referenced, ranging from a lightweight dictionary of the most recent rankings containing only the team abbreviation and rank, to a much larger dictionary of all rankings for an entire season with results including full team name and abbreviation, current rank, week number, previous rank, and movement.

```
from sportsreference.ncaab.rankings import Rankings

rankings = Rankings()
# Prints a dictionary of just the team abbreviation and rank for the current
# week
print(rankings.current)
# Prints more detailed information including previous rank, full name, and
# movement for all teams in current week
print(rankings.current_extended)
# Prints detailed information for all teams for all weeks where rankings
# have been published for the requested season.
print(rankings.complete)
```

**class** `sportsreference.ncaab.rankings.Rankings` (`year=None`)

Bases: `object`

Get all Associated Press (AP) rankings on a week-by-week basis.

Grab a list of the rankings published by the Associated Press to easily query the hierarchy of teams each week. The results expose the current and previous rankings as well as the movement for each team in the list.



**Parameters** `year` (*string (optional)*) – A string of the requested year to pull rankings from. Defaults to the most recent season.

#### **complete**

Returns a dictionary where each key is a week number as an `int` and each value is a list of dictionaries containing the AP rankings for each week. Within each list is a dictionary of team information such as name, abbreviation, rank, and more. Note that the list might not necessarily be in the same order as the rankings.

The overall dictionary has the following structure:

```
{
  week number, ie 19 (int): [
    {
      'abbreviation': Team's abbreviation, such as 'PURDUE'
                        (str),
      'name': Team's full name, such as 'Purdue' (str),
      'rank': Team's rank for the current week (int),
      'week': Week number for the results, such as 19 (int),
      'date': Date the rankings were released, such as
              '2017-03-01'. Can also be 'Final' for the final
              rankings or 'Preseason' for preseason rankings
              (str),
      'previous': The team's previous rank, if applicable
                  (str),
      'change': The amount the team moved up or down the
                 rankings. Moves up the ladder have a positive
                 number while drops yield a negative number
                 and teams that didn't move have 0 (int)
    },
    ...
  ],
  ...
}
```

#### **current**

Returns a dictionary of the most recent rankings from the Associated Press where each key is a string of the team's abbreviation and each value is an `int` of the team's rank for the current week.

#### **current\_extended**

Returns a list of dictionaries of the most recent AP rankings. The list is ordered in terms of the ranking so the #1 team will be in the first element and the #25 team will be the last element. Each dictionary has the following structure:

```
{
  'abbreviation': Team's abbreviation, such as 'PURDUE' (str),
  'name': Team's full name, such as 'Purdue' (str),
  'rank': Team's rank for the current week (int),
  'week': Week number for the results, such as 19 (int),
  'date': Date the rankings were released, such as '2017-03-01'.
          Can also be 'Final' for the final rankings or
          'Preseason' for preseason rankings (str),
  'previous': The team's previous rank, if applicable (str),
  'change': The amount the team moved up or down the rankings.
             Moves up the ladder have a positive number while
             drops yield a negative number and teams that didn't
             move have 0 (int)
}
```

## Roster

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the `Player` class which has detailed information ranging from career points totals to single-season stats and player height and weight. The following is an example on collecting career information for Carsen Edwards.

```
from sportsreference.ncaab.roster import Player

carsen_edwards = Player('carsen-edwards-1')
print(carsen_edwards.name)  # Prints 'Carsen Edwards'
print(carsen_edwards.points)  # Prints Edwards' career points total
# Prints a Pandas DataFrame of all relevant stats per season for Edwards
print(carsen_edwards.dataframe)
```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific season, call the class instance with the season string. All future property requests will return the season-specific stats.

```
from sportsreference.ncaab.roster import Player

carsen_edwards = Player('carsen-edwards-1')  # Currently pulling career stats
print(carsen_edwards.points)  # Prints Edwards' CAREER points total
# Prints Edwards' points total only for the 2017-18 season.
print(carsen_edwards('2017-18').points)
# Prints the number of games Edwards played in the 2017-18 season.
print(carsen_edwards.games_played)
```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.ncaab.roster import Player

carsen_edwards = Player('carsen-edwards-1')  # Currently pulling career stats
# Prints Edwards' points total only for the 2017-18 season.
print(carsen_edwards('2017-18').points)
print(carsen_edwards('Career').points)  # Prints Edwards' career points total
```

In addition, the Roster module also contains the `Roster` class which can be used to pull all players on a team's roster during a given season and creates instances of the `Player` class for each team member and adds them to a list to be easily queried.

```
from sportsreference.ncaab.roster import Roster

purdue = Roster('PURDUE')
for player in purdue.players:
    # Prints the name of all players who played for Purdue in the most
    # recent season.
    print(player.name)
```

**class** `sportsreference.ncaab.roster.Player` (*player\_id*)  
Bases: `sportsreference.ncaab.player.AbstractPlayer`

Get player information and stats for all seasons.

Given a player ID, such as 'carsen-edwards-1' for Carsen Edwards, capture all relevant stats and information like name, height/weight, career three-pointers, last season's offensive rebounds, offensive points contributed, and much more.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [sports-reference.com](https://sports-reference.com).

**Parameters** `player_id` (*string*) – A player's ID according to [sports-reference.com](https://sports-reference.com), such as 'carsen-edwards-1' for Carsen Edwards. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'first-last-N' where 'first' is the player's first name in lowercase, 'last' is the player's last name in lowercase, and 'N' is a number starting at '1' for the first time that player ID has been used and increments by 1 for every successive player.

**box\_plus\_minus**

Returns a `float` of the total number of points per 100 possessions the player contributed in comparison to an average player in the league.

**conference**

Returns a `string` of the abbreviation for the conference the team participated in for the requested season.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values where each index is a different season plus the career stats.

**defensive\_box\_plus\_minus**

Returns a `float` of the number of defensive points per 100 possessions the player contributed in comparison to an average player in the league.

**defensive\_win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his defensive plays.

**games\_played**

Returns an `int` of the number of games the player participated in.

**games\_started**

Returns an `int` of the number of games the player started.

**height**

Returns a `string` of the player's height in the format "feet-inches".

**offensive\_box\_plus\_minus**

Returns a `float` of the number of offensive points per 100 possessions the player contributed in comparison to an average player in the league.

**offensive\_win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his offensive plays.

**player\_efficiency\_rating**

Returns a `float` of the player's efficiency rating which represents the player's relative production level. An average player in the league has an efficiency rating of 15.

**points\_produced**

Returns an `int` of the number of offensive points the player produced.

**position**

Returns a `string` constant of the player's primary position.

**season**

Returns a `string` of the season in the format 'YYYY-YY', such as '2017-18'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**team\_abbreviation**

Returns a `string` of the abbreviation for the team the player plays for, such as 'PURDUE' for Carsen Edwards.

**weight**

Returns an `int` of the player's weight in pounds.

**win\_shares**

Returns a `float` of the number of wins the player contributed to the team as a result of his offensive and defensive plays.

**win\_shares\_per\_40\_minutes**

Returns a `float` of the number of wins the player contributed to the team per 40 minutes of playtime. An average player has a contribution of 0.100.

**class** `sportsreference.ncaab.roster.Roster` (*team*, *year=None*, *slim=False*)

Bases: `object`

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the `Player` class for each player, containing a detailed list of the players statistics and information.

**Parameters**

- **team** (*string*) – The team's abbreviation, such as 'PURDUE' for the Purdue Boilermakers.
- **year** (*string (optional)*) – The 4-digit year to pull the roster from, such as '2018'. If left blank, defaults to the most recent season.
- **slim** (*boolean (optional)*) – Set to `True` to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to `False`.

**players**

Returns a `list` of player instances for each player on the requested team's roster if the `slim` property is `False` when calling the `Roster` class. If the `slim` property is `True`, returns a `dictionary` where each key is a `string` of the player's ID and each value is the player's first and last name as listed on the roster page.

## Schedule

The `Schedule` module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.ncaab.schedule import Schedule

purdue_schedule = Schedule('PURDUE')
for game in purdue_schedule:
    print(game.date)    # Prints the date the game was played
    print(game.result)  # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** `sportsreference.ncaab.schedule.Game` (*game\_data*)

Bases: `object`

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

**Parameters** `game_data` (*string*) – The row containing the specified game information.

**arena**

Returns a *string* of the name of the arena the game was played at.

**boxscore**

Returns an instance of the Boxscore class containing more detailed stats on the game.

**boxscore\_index**

Returns a *string* of the URI for a boxscore which can be used to access or index a game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame representing the Boxscore class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property. The index for the DataFrame is the boxscore string.

**date**

Returns a *string* of the game's date, such as 'Fri, Nov 10, 2017'.

**datetime**

Returns a datetime object to indicate the month, day, year, and time the requested game took place.

**game**

Returns an *int* of the game's position in the season. The first game of the season returns 1.

**location**

Returns a *string* constant to indicate whether the game was played at the team's home venue, the opponent's venue, or at a neutral site.

**opponent\_abbr**

Returns a *string* of the opponent's abbreviation, such as 'PURDUE' for the Purdue Boilermakers.

**opponent\_conference**

Returns a *string* of the opponent's conference, such as 'Big Ten' for a team participating in the Big Ten Conference. If the team is not a Division-I school, a string constant for non-majors is returned.

**opponent\_name**

Returns a *string* of the opponent's name, such as the 'Purdue Boilermakers'.

**opponent\_rank**

Returns a *string* of the opponent's rank when the game was played and None if the team was unranked.

**overtimes**

Returns an *int* of the number of overtimes that were played during the game and 0 if the game finished at the end of regulation time.

**points\_against**

Returns an *int* of the number of points the team allowed during the game.

**points\_for**

Returns an *int* of the number of points the team scored during the game.

**result**

Returns a *string* constant to indicate whether the team won or lost the game.

**season\_losses**

Returns an `int` of the number of games the team has lost after the conclusion of the requested game.

**season\_wins**

Returns an `int` of the number of games the team has won after the conclusion of the requested game.

**streak**

Returns a `string` of the team's win streak at the conclusion of the requested game. Streak is in the format '[W/L] #' (ie. 'W 3' indicates a 3-game winning streak while 'L 2' indicates a 2-game losing streak.

**time**

Returns a `string` to indicate the time the game started, such as '9:00 pm/est'.

**type**

Returns a `string` constant to indicate whether the game was played during the regular season or in the post season.

**class** `sportsreference.ncaab.schedule.Schedule` (*abbreviation*, *year=None*)

Bases: `object`

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

**Parameters**

- **abbreviation** (*string*) – A team's short name, such as 'PURDUE' for the Purdue Boilermakers.
- **year** (*string* (*optional*)) – The requested year to pull stats from.

**dataframe**

Returns a pandas DataFrame where each row is a representation of the Game class. Rows are indexed by the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame where each row is a representation of the Boxscore class for every game in the schedule. Rows are indexed by the boxscore string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The Teams module exposes information for all NCAAB teams including the team name and abbreviation, the number of games they won during the season, the total number of shots they've blocked, and much more.

```
from sportsreference.ncaab.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    print(team.blocks)  # Prints the number of shots the team blocked
```

Each Team instance contains a link to the `Schedule` class which enables easy iteration over all games for a particular team. A Pandas DataFrame can also be queried to easily grab all stats for all games.

```
from sportsreference.ncaab.teams import Teams

teams = Teams()
for team in teams:
```

(continues on next page)

(continued from previous page)

```

schedule = team.schedule # Returns a Schedule instance for each team
# Returns a Pandas DataFrame of all metrics for all game Boxscores for
# a season.
df = team.schedule.dataframe_extended

```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```

from sportsreference.ncaab.teams import Teams

for team in Teams():
    roster = team.roster # Gets each team's roster
    for player in roster.players:
        print(player.name) # Prints each players name on the roster

```

```

class sportsreference.ncaab.teams.Team(team_data, team_conference=None, year=None)
    Bases: object

```

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as full and short names, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **team\_conference** (*string (optional)*) – A string of the team's conference abbreviation, such as 'big-12'.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a *string* of the team's short name, such as 'PURDUE' for the Purdue Boilermakers.

#### assist\_percentage

Returns a *float* of the percentage of field goals that were assisted. Percentage ranges from 0-100.

#### assists

Returns an *int* of the total number of assists during the season.

#### away\_losses

Returns an *int* of the total number of away games the team lost during the season.

#### away\_wins

Returns an *int* of the total number of away games the team won during the season.

#### block\_percentage

Returns a *float* of the percentage of 2-point field goals by the opponent that were blocked. Percentage ranges from 0-100.

#### blocks

Returns an *int* of the total number of blocks during the season.

#### conference

Returns a *string* of the team's conference abbreviation, such as 'big-12' for the Big 12 Conference.

#### conference\_losses

Returns an *int* of the total number of conference games the team lost during the season.

**conference\_wins**

Returns an `int` of the total number of conference games the team won during the season.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'PURDUE'.

**defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds during the season.

**effective\_field\_goal\_percentage**

Returns a `float` of the field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts during the season.

**field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts. Percentage ranges from 0-1.

**field\_goals**

Returns an `int` of the total number of field goals made during the season.

**free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt.

**free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts during the season.

**free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts during the season.

**free\_throws**

Returns an `int` of the total number of free throws made during the season.

**free\_throws\_per\_field\_goal\_attempt**

Returns a `float` of the number of free throws per field goal attempt.

**games\_played**

Returns an `int` of the total number of games the team has played during the season.

**home\_losses**

Returns an `int` of the total number of home games the team lost during the season.

**home\_wins**

Returns an `int` of the total number of home games the team won during the season.

**losses**

Returns an `int` of the total number of games the team lost during the season.

**minutes\_played**

Returns an `int` of the total number of minutes played by the team during the season.

**name**

Returns a `string` of the team's full name, such as 'Purdue Boilermakers'.

**net\_rating**

Returns a `float` of the net team rating which is equivalent to the difference between the offensive rating and the defensive (or the opponent's offensive) rating. Positive values indicate teams that score more points than they allow per 100 possessions.



**offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions.

**offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds a team grabbed. Percentage ranges from 0-100.

**offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds during the season.

**opp\_assist\_percentage**

Returns a `float` of the percentage of the opponent's field goals that were assisted. Percentage ranges from 0-100.

**opp\_assists**

Returns an `int` of the total number of assists during the season by opponents.

**opp\_block\_percentage**

Returns a `float` of the percentage of 2-point field goals that were blocked by the opponent. Percentage ranges from 0-100.

**opp\_blocks**

Returns an `int` of the total number of blocks during the season by opponents.

**opp\_defensive\_rebounds**

Returns an `int` of the total number of defensive rebounds during the season by opponents.

**opp\_effective\_field\_goal\_percentage**

Returns a `float` of the opponent's field goal percentage while giving extra weight to 3-point field goals. Percentage ranges from 0-1.

**opp\_field\_goal\_attempts**

Returns an `int` of the total number of field goal attempts during the season by opponents.

**opp\_field\_goal\_percentage**

Returns a `float` of the number of field goals made divided by the total number of field goal attempts by opponents. Percentage ranges from 0-1.

**opp\_field\_goals**

Returns an `int` of the total number of field goals made during the season by opponents.

**opp\_free\_throw\_attempt\_rate**

Returns a `float` of the average number of free throw attempts per field goal attempt by the opponent.

**opp\_free\_throw\_attempts**

Returns an `int` of the total number of free throw attempts during the season by opponents.

**opp\_free\_throw\_percentage**

Returns a `float` of the number of free throws made divided by the number of free throw attempts during the season by opponents.

**opp\_free\_throws**

Returns an `int` of the total number of free throws made during the season by opponents.

**opp\_free\_throws\_per\_field\_goal\_attempt**

Returns a `float` of the number of free throws per field goal attempt by the opponent.

**opp\_offensive\_rating**

Returns a `float` of the average number of points scored per 100 possessions by the opponent. This is equivalent to the team's defensive rating as it is the number of points the team allows per 100 possessions by the opponent.

**opp\_offensive\_rebound\_percentage**

Returns a `float` of the percentage of available offensive rebounds the opponent grabbed. Percentage ranges from 0-100.

**opp\_offensive\_rebounds**

Returns an `int` of the total number of offensive rebounds during the season by opponents.

**opp\_personal\_fouls**

Returns an `int` of the total number of personal fouls during the season by opponents.

**opp\_points**

Returns an `int` of the total number of points opponents have scored during the season.

**opp\_steal\_percentage**

Returns a `float` of the percentage of possessions that ended in a steal by the opponent. Percentage ranges from 0-100.

**opp\_steals**

Returns an `int` of the total number of steals during the season by opponents.

**opp\_three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range by the opponent. Percentage ranges from 0-1.

**opp\_three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts during the season by opponents.

**opp\_three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts by opponents. Percentage ranges from 0-1.

**opp\_three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made during the season by opponents.

**opp\_total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds the opponent grabbed. Percentage ranges from 0-100.

**opp\_total\_rebounds**

Returns an `int` of the total number of rebounds during the season by opponents.

**opp\_true\_shooting\_percentage**

Returns a `float` of the opponent's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**opp\_turnover\_percentage**

Returns a `float` of the number of times the opponent turned the ball over per 100 possessions.

**opp\_turnovers**

Returns an `int` of the total number of turnovers during the season by opponents.

**opp\_two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts during the season by opponents.

**opp\_two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts by opponents. Percentage ranges from 0-1.

**opp\_two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made during the season by opponents.

**pace**

Returns a `float` of the average number of possessions per 40 minutes.

**personal\_fouls**

Returns an `int` of the total number of personal fouls during the season.

**points**

Returns an `int` of the total number of points the team scored during the season.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**schedule**

Returns an instance of the Schedule class containing the team's complete schedule for the season.

**simple\_rating\_system**

Returns a `float` of the team's average point differential compared to the strength of schedule. Higher values indicate stronger teams. An average team is denoted with 0.0. Negative numbers are comparatively worse than average.

**steal\_percentage**

Returns a `float` of the percentage of opponent possessions that ended in a steal. Percentage ranges from 0-100.

**steals**

Returns an `int` of the total number of steals during the season.

**strength\_of\_schedule**

Returns a `float` of the team's strength of schedule based on the points above and below average. An average strength of schedule is denoted with 0.0. Negative numbers are comparatively easier than average.

**three\_point\_attempt\_rate**

Returns a `float` of the percentage of field goal attempts from 3-point range. Percentage ranges from 0-1.

**three\_point\_field\_goal\_attempts**

Returns an `int` of the total number of three point field goal attempts during the season.

**three\_point\_field\_goal\_percentage**

Returns a `float` of the number of three point field goals made divided by the number of three point field goal attempts. Percentage ranges from 0-1.

**three\_point\_field\_goals**

Returns an `int` of the total number of three point field goals made during the season.

**total\_rebound\_percentage**

Returns a `float` of the percentage of available rebounds a team grabbed. Percentage ranges from 0-100.

**total\_rebounds**

Returns an `int` of the total number of rebounds during the season.

**true\_shooting\_percentage**

Returns a `float` of the team's true shooting percentage which considers free throws, 2-point field goals, and 3-point field goals. Percentage ranges from 0-1.

**turnover\_percentage**

Returns a `float` of the number of times the team turned the ball over per 100 possessions.

**turnovers**

Returns an `int` of the total number of turnovers during the season.

**two\_point\_field\_goal\_attempts**

Returns an `int` of the total number of two point field goal attempts during the season.

**two\_point\_field\_goal\_percentage**

Returns a `float` of the number of two point field goals made divided by the number of two point field goal attempts. Percentage ranges from 0-1.

**two\_point\_field\_goals**

Returns an `int` of the total number of two point field goals made during the season.

**win\_percentage**

Returns a `float` of the number of wins divided by the number of games played during the season. Percentage ranges from 0-1.

**wins**

Returns an `int` of the total number of games the team won during the season.

**class** `sportsreference.ncaab.teams.Teams` (*year=None*)

Bases: `object`

A list of all NCAA Men's Basketball teams and their stats in a given year.

Finds and retrieves a list of all NCAA Men's Basketball teams from [www.sports-reference.com](http://www.sports-reference.com) and creates a Team instance for every team that participated in the league in a given year. The Team class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** `year` (*string optional*) – The requested year to pull stats from.

**dataframes**

Returns a pandas DataFrame where each row is a representation of the Team class. Rows are indexed by the team abbreviation.

### 1.6.1.4 NCAAF Package

The NCAAF package offers multiple modules which can be used to retrieve information and statistics for Division-I College Football, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of points scored to the number of pass yards, to the yards from penalties and much more. The Boxscore can be easily queried by passing a boxscore's URI on [sports-reference.com](http://sports-reference.com) which can be retrieved from the Schedule class (see Schedule module below for more information on retrieving game-specific information).

```
from sportsreference.ncaaf.boxscore import Boxscore

game_data = Boxscore('2018-01-08-georgia')
print(game_data.home_points) # Prints 23
print(game_data.away_points) # Prints 26
df = game_data.dataframe # Returns a Pandas DataFrame of game metrics
```

The Boxscore module also contains a `Boxscores` class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.

```
from datetime import datetime
from sportsreference.ncaaf.boxscore import Boxscores

games_today = Boxscores(datetime.today())
print(games_today.games) # Prints a dictionary of all matchups for today
```

The `Boxscores` class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```
from datetime import datetime
from sportsreference.ncaaf.boxscore import Boxscores

# Pulls all games between and including August 30, 2017 and August 31, 2017
games = Boxscores(datetime(2017, 8, 30), datetime(2017, 8, 31))
# Prints a dictionary of all results from August 30, 2017 and August 31,
# 2017
print(games.games)
```

**class** `sportsreference.ncaaf.boxscore.Boxscore(uri)`

Bases: `object`

Detailed information about the final statistics for a game.

Stores all relevant information for a game such as the date, time, location, result, and more advanced metrics such as the number of fumbles from sacks, a team's passing completion, rushing touchdowns and much more.

**Parameters** `uri` (*string*) – The relative link to the boxscore HTML page, such as '2018-01-08-georgia'.

**away\_first\_downs**

Returns an `int` of the number of first downs the away team gained.

**away\_fumbles**

Returns an `int` of the number of times the away team fumbled the ball.

**away\_fumbles\_lost**

Returns an `int` of the number of times the away team turned the ball over as the result of a fumble.

**away\_interceptions**

Returns an `int` of the number of interceptions the away team threw.

**away\_pass\_attempts**

Returns an `int` of the number of passes that were thrown by the away team.

**away\_pass\_completions**

Returns an `int` of the number of completed passes the away team made.

**away\_pass\_touchdowns**

Returns an `int` of the number of passing touchdowns the away team scored.

**away\_pass\_yards**

Returns an `int` of the number of passing yards the away team gained.

**away\_penalties**

Returns an `int` of the number of penalties called on the away team.

**away\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the away team.

**away\_points**

Returns an `int` of the number of points the away team scored.

**away\_rush\_attempts**

Returns an `int` of the number of rushing plays the away team made.

**away\_rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the away team scored.

**away\_rush\_yards**

Returns an `int` of the number of rushing yards the away team gained.

**away\_total\_yards**

Returns an `int` of the total number of yards the away team gained.

**away\_turnovers**

Returns an `int` of the number of times the away team turned the ball over.

**away\_yards\_from\_penalties**

Returns an `int` of the number of yards gifted as a result of penalties called on the away team.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string URI that is used to instantiate the class, such as '2018-01-08-georgia'.

**date**

Returns a `string` of the date the game took place.

**home\_first\_downs**

Returns an `int` of the number of first downs the home team gained.

**home\_fumbles**

Returns an `int` of the number of times the home team fumbled the ball.

**home\_fumbles\_lost**

Returns an `int` of the number of times the home team turned the ball over as the result of a fumble.

**home\_interceptions**

Returns an `int` of the number of interceptions the home team threw.

**home\_pass\_attempts**

Returns an `int` of the number of passes that were thrown by the home team.

**home\_pass\_completions**

Returns an `int` of the number of completed passes the home team made.

**home\_pass\_touchdowns**

Returns an `int` of the number of passing touchdowns the home team scored.

**home\_pass\_yards**

Returns an `int` of the number of passing yards the home team gained.

**home\_penalties**

Returns an `int` of the number of penalties called on the home team.

**home\_players**

Returns a `list` of `BoxscorePlayer` class instances for each player on the home team.

**home\_points**

Returns an `int` of the number of points the home team scored.

**home\_rush\_attempts**

Returns an `int` of the number of rushing plays the home team made.

**home\_rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the home team scored.

**home\_rush\_yards**

Returns an `int` of the number of rushing yards the home team gained.

**home\_total\_yards**

Returns an `int` of the total number of yards the home team gained.

**home\_turnovers**

Returns an `int` of the number of times the home team turned the ball over.

**home\_yards\_from\_penalties**

Returns an `int` of the number of yards gifted as a result of penalties called on the home team.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'GEORGIA' for the Georgia Bulldogs.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Georgia'.

**stadium**

Returns a `string` of the name of the stadium where the game was played.

**time**

Returns a `string` of the time the game started.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'ALABAMA' for the Alabama Crimson Tide.

**winning\_name**

Returns a `string` of the winning team's name, such as 'Alabama'.

```
class sportsreference.ncaaf.boxscore.BoxscorePlayer(player_id,           player_name,
                                                    player_data)
```

Bases: `sportsreference.ncaaf.player.AbstractPlayer`

Get player stats for an individual game.

Given a player ID, such as 'david-blough-1' for David Blough, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

**Parameters**

- **player\_id** (*string*) – A player's ID according to sports-reference.com, such as 'david-blough-1' for David Blough. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'first-last-n' where 'first' is the player's first name, 'last' is the player's last name, and 'n' is a number starting at 1 for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'David Blough'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**average\_kickoff\_return\_yards**

Returns a `float` of the average number of yards the player gained per attempted kickoff return.

**average\_punt\_return\_yards**

Returns a `float` of the average number of yards the player gained per attempted punt return.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and value for the specified game.

**extra\_point\_percentage**

Returns a `float` of the percentage of attempted extra points the player made. Percentage ranges from 0-100.

**extra\_points\_attempted**

Returns an `int` of the total number of extra points the player attempted.

**field\_goal\_percentage**

Returns a `float` of the percentage of attempted field goals the player made. Percentage ranges from 0-100.

**field\_goals\_attempted**

Returns an `int` of the total number of field goals the player attempted.

**kickoff\_return\_yards**

Returns an `int` of the total number of yards the player gained while the player attempted to return a kickoff.

**kickoff\_returns**

Returns an `int` of the number of kickoffs the player attempted to return.

**pass\_yards\_per\_attempt**

Returns a `float` of the average number of yards the player gained per pass attempt.

**points\_kicking**

Returns an `int` of the total number of points the player gained from kicking field goals or extra points.

**punt\_return\_yards**

Returns an `int` of the total number of yards the player gained while the player attempted to return a punt.

**punt\_returns**

Returns an `int` of the number of punts the player attempted to return.

**punting\_yards**

Returns an `int` of the total number of yards the player punted the ball.

**punting\_yards\_per\_attempt**

Returns a `float` of the average number of yards the player punted per attempt.

**punts**

Returns an `int` of the number of times the player punted the ball.

**class** sportsreference.ncaaf.boxscore.Boxscores (*date*, *end\_date=None*)

Bases: `object`

Search for NCAAAF games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, a boolean value which indicates if the game is between two Division-I teams or not, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

**Parameters**

- **date** (*datetime object*) – The date to search for any matches. The month, day, and year are required for the search, but time is not factored into the search.



- **end\_date** (*datetime object*) – Optionally specify an end date to iterate until. All boxscores starting from the date specified in the ‘date’ parameter up to and including the boxscores specified in the ‘end\_date’ parameter will be pulled. If left empty, or if ‘end\_date’ is prior to ‘date’, only the games from the day specified in the ‘date’ parameter will be saved.

### games

Returns a dictionary object representing all of the games played on the requested day. Dictionary is in the following format:

```
{ 'date' : [ # 'date' is the string date in format 'MM-DD-YYYY'
    {
        'home_name': Name of the home team, such as 'Purdue
                    Boilermakers' (`str`),
        'home_abbr': Abbreviation for the home team, such as
                    'PURDUE' (`str`),
        'away_name': Name of the away team, such as 'Indiana
                    Hoosiers' (`str`),
        'away_abbr': Abbreviation for the away team, such as
                    'INDIANA' (`str`),
        'boxscore': String representing the boxscore URI, such as
                    '2018-01-28-15-indiana' (`str`),
        'non_di': Boolean value which evaluates to True when at
                    least one of the teams does not compete in NCAA
                    Division-I basketball (`bool`),
        'top_25': Boolean value which evaluates to True when at
                    least one of the teams is ranked in the AP Top 25
                    polls (`bool`),
        'winning_name': Full name of the winning team, such as
                    'Purdue Boilermakers' (`str`),
        'winning_abbr': Abbreviation for the winning team, such as
                    'PURDUE' (`str`),
        'losing_name': Full name of the losing team, such as
                    'Indiana Hoosiers' (`str`),
        'losing_abbr': Abbreviation for the losing team, such as
                    'INDIANA' (`str`),
        'home_score': Integer score for the home team (`int`),
        'home_rank': Integer representing the home team's rank
                    (`int`),
        'away_score': Integer score for the away team (`int`),
        'away_rank': Integer representing the away team's rank
                    (`int`)
    },
    { ... },
    ...
  ]
}
```

If no games were played on ‘date’, the list for [‘date’] will be empty.

sportsreference.ncaaf.boxscore.ncaaf\_int\_property\_sub\_index (*func*)

## Conferences

The Conference module allows conferences to be pulled for any season using the `Conferences` class. Accessing the class properties exposes various dictionaries containing the team and conference abbreviations as well as other information. To get a list of conference abbreviations for each team, query the `team_conference` property.

```
from sportsreference.ncaaf.conferences import Conferences

conferences = Conferences()
# Prints a dictionary of the team abbreviation as a key and conference
# abbreviation as the value.
print(conferences.team_conference)
```

The conferences property can also be queried to provide more details on the teams in every conference.

```
from sportsreference.ncaab.conferences import Conferences

conferences = Conferences()
# Prints a dictionary where each key is the conference abbreviation and
# each value is a dictionary containing the full conference name as well as
# another dictionary of all teams in the conference, including name and
# abbreviation for each team.
print(conferences.conferences)
```

```
class sportsreference.ncaaf.conferences.Conference (conference_abbreviation,
                                                    year=None,                ig-
                                                    ignore_missing=False)
```

Bases: object

Find teams that participated in a particular conference.

Create a dictionary which includes the names and abbreviations for all teams that participated in a conference during a given year.

#### Parameters

- **conference\_abbreviation** (*string*) – A string of the requested conference’s abbreviation, such as ‘big-12’.
- **year** (*string (optional)*) – A string of the requested year to pull conference information from. Defaults to the most recent season.
- **ignore\_missing** (*boolean (optional)*) – A boolean which, when True, doesn’t throw an error if the requested conference has an incomplete or empty page on [www.sports-reference.com](http://www.sports-reference.com), preventing the parsing from completing successfully.

#### teams

Returns a dictionary of team names and abbreviations where each key is a string of the team abbreviation and each value is a string of the full team name.

```
class sportsreference.ncaaf.conferences.Conferences (year=None,                ig-
                                                    ignore_missing=False)
```

Bases: object

Get all conferences and teams for a season.

Retrieve a list of all conferences and teams that participated in the conference for each team in the season. The included properties allow flexibility in queries to either get the conference abbreviation for a given team, or get more detailed information including all teams for each conference.

#### Parameters

- **year** (*string (optional)*) – A string of the requested year to pull conferences from. Defaults to the most recent season.
- **ignore\_missing** (*boolean (optional)*) – A boolean which, when True, doesn’t throw an error if the any conference has an incomplete or empty page on [www.sports-reference.com](http://www.sports-reference.com), preventing the parsing from completing successfully.

**conferences**

Returns a dictionary of conference names and abbreviations where each key is a string of the abbreviation and each value is a dictionary containing the full conference name and another dictionary with individual team information. The overall dictionary is in the following structure:

```
{
  abbreviation, ie 'big-12' (str): {
    'name': Full conference name, such as 'Big 12 Conference'
        (str),
    'teams': {
      team abbreviation, such as 'kansas' (str): Full team
        name, such as 'Kansas' (str),
      ...
    }
  },
  ...
}
```

**team\_conference**

Returns a dictionary of conference abbreviations for each team where each key is a string of the team abbreviation and each value is a string of the conference abbreviation.

**Player**

The Player module contains an abstract base class that can be inherited by both the BoxscorePlayer and Player classes in the Boxscore and Roster modules, respectively. All of the properties that appear in the AbstractPlayer class can be read from either of the two child classes mentioned above.

```
class sportsreference.ncaaf.player.AbstractPlayer (player_id,          player_name,
                                                    player_data)
```

Bases: object

Get player information and stats for all seasons.

Given a player ID, such as 'david-blough-1' for David Blough, capture all relevant stats and information like name, team, height/weight, career starts, single season passing yards, sacks, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on sports-reference.com.

**Parameters**

- **player\_id** (*string*) – A player's ID according to sports-reference.com, such as 'david-blough-1' for David Blough. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'first-last-n' where 'first' is the player's first name in lowercase, 'last' is the player's last name in lowercase, and 'n' is a number starting at '1' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'David Blough'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**adjusted\_yards\_per\_attempt**

Returns a float of the adjusted number of yards gained per passing attempt, equal to (yards + 20 \* pass\_touchdowns - 45 \* interceptions) / pass\_attempts.

**assists\_on\_tackles**

Returns an `int` of the number of assists the player made on tackles.

**attempted\_passes**

Returns an `int` of the number of passes the player attempted.

**completed\_passes**

Returns an `int` of the number of completed passes the player threw.

**extra\_points\_made**

Returns an `int` of the number of extra points the player made.

**field\_goals\_made**

Returns an `int` of the total number of field goals the player made from any distance.

**fumbles\_forced**

Returns an `int` of the number of times the player forced a fumble.

**fumbles\_recovered**

Returns an `int` of the number of fumbles the player has recovered.

**fumbles\_recovered\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after recovering a fumble.

**interceptions**

Returns an `int` of the number of times the player intercepted a pass.

**interceptions\_returned\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after intercepting a pass. Commonly referred to as a 'Pick-6'.

**interceptions\_thrown**

Returns an `int` of the number of interceptions the player has thrown.

**kickoff\_return\_touchdowns**

Returns an `int` of the number of kickoffs the player returned for a touchdown.

**name**

Returns a `string` of the player's name, such as 'David Blough'.

**pass\_attempts**

Returns an `int` of the number of passes the player attempted.

**passes\_defended**

Returns an `int` of the number of passes the player has defended as a defensive player.

**passing\_completion**

Returns a `float` of the percentage of passes that were caught by a receiver. Percentage ranges from 0-100.

**passing\_touchdowns**

Returns an `int` of the number of touchdowns passes the player has thrown.

**passing\_yards**

Returns an `int` of the total number of yards the player gained from passing the ball.

**passing\_yards\_per\_attempt**

Returns a `float` of the number of yards gained per passing attempt.

**player\_id**

Returns a `string` of the player's ID on sports-reference, such as 'david-blough-1' for David Blough.

**plays\_from\_scrimmage**

Returns an `int` of the combined number of rushing attempts and receptions the player had.

**punt\_return\_touchdowns**

Returns an `int` of the number of punts the player returned for a touchdown.

**quarterback\_rating**

Returns a `float` of the player's quarterback rating.

**receiving\_touchdowns**

Returns an `int` of the number of touchdowns the player scored after receiving a pass.

**receiving\_yards**

Returns an `int` of the number of receiving yards the player gained.

**receiving\_yards\_per\_reception**

Returns a `float` of the average number of yards the player gained per reception.

**receptions**

Returns an `int` of the number of receptions the player made.

**rush\_attempts**

Returns an `int` of the number of rushing plays the player attempted.

**rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the player scored.

**rush\_yards**

Returns an `int` of the number of rushing yards the player gained.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing attempt.

**rushing\_and\_receiving\_touchdowns**

Returns an `int` of the combined number of rushing and receiving touchdowns the player scored.

**sacks**

Returns a `float` of the number of times the player sacked a quarterback.

**solo\_tackles**

Returns an `int` of the number of tackles the player made by himself.

**tackles\_for\_loss**

Returns a `float` of the number of tackles for a loss the player made.

**total\_tackles**

Returns an `int` of the number of tackles the player made.

**yards\_from\_scrimmage**

Returns an `int` of the total number of yards gained from scrimmage for both rushing and receiving.

**yards\_from\_scrimmage\_per\_play**

Returns a `float` of the average number of yards gained per rushing attempt and/or reception.

**yards\_recovered\_from\_fumble**

Returns an `int` of the number of yards the player gained after recovering a fumble.

**yards\_returned\_from\_interceptions**

Returns an `int` of the number of yards the player returned after intercepting a pass.

**yards\_returned\_per\_interception**

Returns a `float` of the average number of yards the player returns after intercepting a pass.

## Rankings

The Rankings module include the `Rankings` class which can be used to easily query the NCAA Men's Division-I Football rankings published by the Associated Press on a week-by-week basis. Different formats can be referenced, ranging from a lightweight dictionary of the most recent rankings containing only the team abbreviation and rank, to a much larger dictionary of all rankings for an entire season with results including full team name and abbreviation, current rank, week number, previous rank, and movement.

```
from sportsreference.ncaaf.rankings import Rankings

rankings = Rankings()
# Prints a dictionary of just the team abbreviation and rank for the current
# week.
print(rankings.current)
# Prints more detailed information including previous rank, full name, and
# movement for all teams for the current week.
print(rankings.current_extended)
# Prints detailed information for all teams for all weeks where rankings
# have been published for the requested season.
print(rankings.complete)
```

**class** `sportsreference.ncaaf.rankings.Rankings` (*year=None*)

Bases: `object`

Get all Associated Press (AP) rankings on a week-by-week basis.

Grab a list of the rankings published by the Associated Press to easily query the hierarchy of teams each week. The results expose the current and previous rankings as well as the movement for each team in the list.

**Parameters** *year* (*string (optional)*) – A string of the requested year to pull rankings from. Defaults to the most recent season.

### **complete**

Returns a dictionary where each key is a week number as an `int` and each value is a list of dictionaries containing the AP rankings for each week. Within each list is a dictionary of team information such as name, abbreviation, rank, and more. Note that the list might not necessarily be in the same order as the rankings.

The overall dictionary has the following structure:

```
{
    week number, ie 16 (int): [
        {
            'abbreviation': Team's abbreviation, such as 'PURDUE'
                           (str),
            'name': Team's full name, such as 'Purdue' (str),
            'rank': Team's rank for the current week (int),
            'week': Week number for the results, such as 16 (int),
            'date': Date the rankings were released, such as
                   '2017-12-03'. Can also be 'Final' for the final
                   rankings or 'Preseason' for preseason rankings
                   (str),
            'previous': The team's previous rank, if applicable
                       (str),
            'change': The amount the team moved up or down the
                     rankings. Moves up the ladder have a positive
                     number while drops yield a negative number
                     and teams that didn't move have 0 (int)
```

(continues on next page)

(continued from previous page)

```

        },
        ...
    ],
    ...
}

```

**current**

Returns a dictionary of the most recent rankings from the Associated Press where each key is a string of the team's abbreviation and each value is an int of the team's rank for the current week.

**current\_extended**

Returns a list of dictionaries of the most recent AP rankings. The list is ordered in terms of the ranking so the #1 team will be in the first element and the #25 team will be the last element. Each dictionary has the following structure:

```

{
    'abbreviation': Team's abbreviation, such as 'PURDUE' (str),
    'name': Team's full name, such as 'Purdue' (str),
    'rank': Team's rank for the current week (int),
    'week': Week number for the results, such as 19 (int),
    'date': Date the rankings were released, such as '2017-03-01'.
            Can also be 'Final' for the final rankings or
            'Preseason' for preseason rankings (str),
    'previous': The team's previous rank, if applicable (str),
    'change': The amount the team moved up or down the rankings.
               Moves up the ladder have a positive number while
               drops yield a negative number and teams that didn't
               move have 0 (int)
}

```

**Roster**

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the Player class which has detailed information ranging from career touchdowns to single-season stats and player height, weight, and nationality. The following is an example on collecting career information for David Blough.

```

from sportsreference.ncaaf.roster import Player

blough = Player('david-blough-1')
print(blough.name)    # Prints 'David Blough'
print(blough.passing_yards)  # Prints Blough's career passing yards
# Prints a Pandas DataFrame of all relevant stats per season for Blough
print(blough.dataframe)

```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific team, call the class instance with the season string. All future property requests will return the season-specific stats.

```

from sportsreference.ncaaf.roster import Player

blough = Player('david-blough-1')  # Currently pulling career stats
print(blough.passing_yards)  # Prints Blough's CAREER passing yards total
# Prints Blough's passing yards total only for the 2017 season
print(blough('2017').passing_yards)
# Prints Blough's passing touchdowns for the 2017 season only
print(blough.passing_touchdowns)

```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.ncaaf.roster import Player

blough = Player('david-blough-1') # Currently pulling career stats
# Prints Blough's passing yards total only for the 2017 season
print(blough('2017').passing_yards)
print(blough('Career').passing_yards) # Prints Blough's career passing yards
```

In addition, the Roster module also contains the Roster class which can be used to pull all players on a team's roster during a given season and creates instances of the Player class for each team member and adds them to a list to be easily queried.

```
from sportsreference.ncaaf.roster import Roster

boilermakers = Roster('PURDUE')
for player in boilermakers.players:
    # Prints the name of all players who played for the Purdue Boilermakers
    # in the most recent season.
    print(player.name)
```

**class** sportsreference.ncaaf.roster.**Player** (*player\_id*)  
Bases: *sportsreference.ncaaf.player.AbstractPlayer*

Get player information and stats for all seasons.

Given a player ID, such as 'david-blough-1' for David Blough, capture all relevant stats and information like name, team, height/weight, career starts, single season passing yards, sacks, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on sports-reference.com.

**Parameters** *player\_id* (*string*) – A player's ID according to sports-reference.com, such as 'david-blough-1' for David Blough. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'first-last-n' where 'first' is the player's first name in lowercase, 'last' is the player's last name in lowercase, and 'n' is a number starting at '1' for the first time that player ID has been used and increments by 1 for every successive player.

**adjusted\_yards\_per\_attempt**

Returns a *float* of the adjusted number of yards gained per passing attempt, equal to  $(yards + 20 * pass\_touchdowns - 45 * interceptions) / pass\_attempts$ .

**assists\_on\_tackles**

Returns an *int* of the number of assists the player made on tackles.

**attempted\_passes**

Returns an *int* of the number of passes the player attempted.

**completed\_passes**

Returns an *int* of the number of completed passes the player threw.

**dataframe**

Returns a *pandas DataFrame* containing all other relevant class properties and values where each index is a different season plus the career stats.

**extra\_points\_made**

Returns an *int* of the number of extra points the player made.



**field\_goals\_made**

Returns an `int` of the total number of field goals the player made from any distance.

**fumbles\_forced**

Returns an `int` of the number of times the player forced a fumble.

**fumbles\_recovered**

Returns an `int` of the number of fumbles the player has recovered.

**fumbles\_recovered\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after recovering a fumble.

**games**

Returns an `int` of the number of games the player participated in.

**height**

Returns a `string` of the player's height in the format "feet-inches".

**interceptions**

Returns an `int` of the number of times the player intercepted a pass.

**interceptions\_returned\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after intercepting a pass. Commonly referred to as a 'Pick-6'.

**interceptions\_thrown**

Returns an `int` of the number of interceptions the player has thrown.

**kickoff\_return\_touchdowns**

Returns an `int` of the number of kickoffs the player returned for a touchdown.

**other\_touchdowns**

Returns an `int` of the total number of all other types of touchdowns the player has scored.

**pass\_attempts**

Returns an `int` of the number of passes the player attempted.

**passes\_defended**

Returns an `int` of the number of passes the player has defended as a defensive player.

**passing\_completion**

Returns a `float` of the percentage of passes that were caught by a receiver. Percentage ranges from 0-100.

**passing\_touchdowns**

Returns an `int` of the number of touchdowns passes the player has thrown.

**passing\_yards**

Returns an `int` of the total number of yards the player gained from passing the ball.

**plays\_from\_scrimmage**

Returns an `int` of the combined number of rushing attempts and receptions the player had.

**points**

Returns an `int` of the number of points the player has scored.

**position**

Returns a `string` of the player's primary position.

**punt\_return\_touchdowns**

Returns an `int` of the number of punts the player returned for a touchdown.

**quarterback\_rating**

Returns a `float` of the player's quarterback rating.

**receiving\_touchdowns**

Returns an `int` of the number of touchdowns the player scored after receiving a pass.

**receiving\_yards**

Returns an `int` of the number of receiving yards the player gained.

**receiving\_yards\_per\_reception**

Returns a `float` of the average number of yards the player gained per reception.

**receptions**

Returns an `int` of the number of receptions the player made.

**rush\_attempts**

Returns an `int` of the number of rushing plays the player attempted.

**rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the player scored.

**rush\_yards**

Returns an `int` of the number of rushing yards the player gained.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing attempt.

**rushing\_and\_receiving\_touchdowns**

Returns an `int` of the combined number of rushing and receiving touchdowns the player scored.

**sacks**

Returns a `float` of the number of times the player sacked a quarterback.

**safeties**

Returns an `int` of the number of safeties the player has scored.

**season**

Returns a `string` of the season in the format 'YYYY', such as '2017'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**solo\_tackles**

Returns an `int` of the number of tackles the player made by himself.

**tackles\_for\_loss**

Returns a `float` of the number of tackles for a loss the player made.

**team\_abbreviation**

Returns a `string` of the team's abbreviation, such as 'PURDUE' for the Purdue Boilermakers.

**total\_tackles**

Returns an `int` of the number of tackles the player made.

**total\_touchdowns**

Returns an `int` of the total number of touchdowns the player has scored.

**two\_point\_conversions**

Returns an `int` of the number of two point conversions the player has scored.

**weight**

Returns an `int` of the player's weight in pounds.

**yards\_from\_scrimmage**

Returns an `int` of the total number of yards gained from scrimmage for both rushing and receiving.

**yards\_from\_scrimmage\_per\_play**

Returns a `float` of the average number of yards gained per rushing attempt and/or reception.

**yards\_recovered\_from\_fumble**

Returns an `int` of the number of yards the player gained after recovering a fumble.

**yards\_returned\_from\_interceptions**

Returns an `int` of the number of yards the player returned after intercepting a pass.

**yards\_returned\_per\_interception**

Returns a `float` of the average number of yards the player returns after intercepting a pass.

**year**

Returns a `string` of the player's class designation, such as 'FR' for freshmen.

**class** sportsreference.ncaaf.roster.**Roster** (*team*, *year=None*, *slim=False*)

Bases: `object`

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the `Player` class for each player, containing a detailed list of the player's statistics and information.

**Parameters**

- **team** (*string*) – The team's abbreviation, such as 'PURDUE' for the Purdue Boilermakers.
- **year** (*string (optional)*) – The 4-digit year to pull the roster from, such as '2017'. If left blank, defaults to the most recent season.
- **slim** (*boolean (optional)*) – Set to `True` to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to `False`.

**players**

Returns a `list` of player instances for each player on the requested team's roster if the `slim` property is `False` when calling the `Roster` class. If the `slim` property is `True`, returns a `dictionary` where each key is a `string` of the player's ID and each value is the player's first and last name as listed on the roster page.

## Schedule

The `Schedule` module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.ncaaf.schedule import Schedule

purdue_schedule = Schedule('PURDUE')
for game in purdue_schedule:
    print(game.date)    # Prints the date the game was played
    print(game.result)  # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** sportsreference.ncaaf.schedule.**Game** (*game\_data*)

Bases: `object`

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

**Parameters** `game_data` (*string*) – The row containing the specified game information.

**boxscore**

Returns an instance of the Boxscore class containing more detailed stats on the game.

**boxscore\_index**

Returns a *string* of the URI for a boxscore which can be used to access or index a game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame representing the Boxscore class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter ‘dataframe’ property. The index for the DataFrame is the boxscore string.

**date**

Returns a *string* of the date the game was played, such as ‘Sep 2, 2017’.

**datetime**

Returns a datetime object of the month, day, year, and time the game was played. If the game doesn’t include a time, the default value of ‘00:00’ will be used.

**day\_of\_week**

Returns a *string* of the 3-letter abbreviation of the day of the week the game was played on, such as ‘Sat’ for Saturday.

**game**

Returns an *int* to indicate which game in the season was requested. The first game of the season returns 1.

**location**

Returns a *string* constant to indicate whether the game was played at home, away, or in a neutral location.

**losses**

Returns an *int* of the number of games the team has lost so far in the season at the conclusion of the requested game.

**opponent\_abbr**

Returns a *string* of the opponent’s abbreviation, such as ‘PURDUE’ for the Purdue Boilermakers.

**opponent\_conference**

Returns a *string* of the conference the team participates in, such as ‘Big Ten’ for the Big Ten Conference. If a team does not compete in Division-I, a string constant for the non-major school will be returned.

**opponent\_name**

Returns a *string* of the opponent’s name, such as ‘Purdue Boilermakers’ for the Purdue Boilermakers.

**opponent\_rank**

Returns an *int* of the opponent’s rank at the time the game was played.

**points\_against**

Returns an *int* of the number of points the team allowed during the game.

**points\_for**

Returns an *int* of the number of points the team scored during the game.

**rank**

Returns an *int* of the team’s rank at the time the game was played.

**result**

Returns a `string` constant to indicate whether the team won or lost the game.

**streak**

Returns a `string` of the team's winning streak at the conclusion of the requested game. Streaks are listed in the format '[W/L] #' (ie. 'W 3' for a 3-game winning streak and 'L 2' for a 2-game losing streak).

**time**

00 PM'.

**Type** Returns a `string` of the time the game started, such as '12

**wins**

Returns an `int` of the number of games the team has won so far in the season at the conclusion of the requested game.

**class** `sportsreference.ncaaf.schedule.Schedule` (*abbreviation*, *year=None*)

Bases: `object`

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

**Parameters**

- **abbreviation** (*string*) – A team's short name, such as 'MICHIGAN' for the Michigan Wolverines.
- **year** (*string* (*optional*)) – The requested year to pull stats from.

**dataframe**

Returns a `pandas DataFrame` where each row is a representation of the `Game` class. Rows are indexed by the `boxscore` string.

**dataframe\_extended**

Returns a `pandas DataFrame` where each row is a representation of the `Boxscore` class for every game in the schedule. Rows are indexed by the `boxscore` string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The `Teams` module exposes information for all NCAAF teams including the team name and abbreviation, the number of games they won during the season, the total number of pass yards, and much more.

```
from sportsreference.ncaaf.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    print(team.pass_yards) # Prints the team's total passing yards
```

Each `Team` instance contains a link to the `Schedule` class which enables easy iteration over all games for a particular team. A `Pandas DataFrame` can also be queried to easily grab all stats for all games.

```
from sportsreference.ncaaf.teams import Teams

teams = Teams()
for team in teams:
    schedule = team.schedule # Returns a Schedule instance for each team
```

(continues on next page)

(continued from previous page)

```
# Returns a Pandas DataFrame of all metrics for all game Boxscores for
# a season.
df = team.schedule.dataframe_extended
```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```
from sportsreference.ncaaf.teams import Teams

for team in Teams():
    roster = team.roster # Gets each team's roster
    for player in roster.players:
        print(player.name) # Prints each players name on the roster
```

**class** sportsreference.ncaaf.teams.Team(team\_data, team\_conference=None, year=None)  
Bases: object

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as full and short names, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **team\_conference** (*string (optional)*) – A string of the team's conference abbreviation, such as 'big-12'.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a *string* of the team's short name, such as 'PURDUE' for the Purdue Boilermakers.

#### conference

Returns a *string* of the team's conference abbreviation, such as 'big-12' for the Big 12 Conference.

#### conference\_losses

Returns an *int* of the total number of conference games the team lost during the season.

#### conference\_win\_percentage

Returns a *float* of the percentage of conference wins divided by the number of conference games played during the season. Percentage ranges from 0-1.

#### conference\_wins

Returns an *int* of the total number of conference games the team won during the season.

#### dataframe

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'PURDUE'.

#### first\_downs

Returns a *float* of the total number of first downs achieved per game.

#### first\_downs\_from\_penalties

Returns a *float* of the average number of first downs from an opponent's penalties per game.

#### fumbles\_lost

Returns a *float* of the average number of fumbles per game.

**games**

Returns an `int` of the total number of games the team has played during the season.

**interceptions**

Returns a `float` of the average number of interceptions thrown per game.

**losses**

Returns an `int` of the total number of games the team lost during the season.

**name**

Returns a `string` of the team's full name, such as 'Purdue Boilermakers'.

**opponents\_first\_downs**

Returns a `float` of the opponents' total number of first downs achieved per game.

**opponents\_first\_downs\_from\_penalties**

Returns a `float` of the opponents' average number of first downs from an opponent's penalties per game.

**opponents\_fumbles\_lost**

Returns a `float` of the opponents' average number of fumbles per game.

**opponents\_interceptions**

Returns a `float` of the opponents' average number of interceptions thrown per game.

**opponents\_pass\_attempts**

Returns a `float` of the opponents' average number of passes that are attempted per game.

**opponents\_pass\_completion\_percentage**

Returns a `float` of the opponents' percentage of completed passes per game. Percentage ranges from 0-100.

**opponents\_pass\_completions**

Returns a `float` of the opponents' average number of completed passes per game.

**opponents\_pass\_first\_downs**

Returns a `float` of the opponents' average number of first downs from passing plays per game.

**opponents\_pass\_touchdowns**

Returns a `float` of the opponents' average number of passing touchdowns scored per game.

**opponents\_pass\_yards**

Returns a `float` of the opponents' average number of yards gained from passing per game.

**opponents\_penalties**

Returns a `float` of the opponents' average number of penalties conceded per game.

**opponents\_plays**

Returns a `float` of the opponents' average number of offensive plays per game.

**opponents\_rush\_attempts**

Returns a `float` of the opponents' average number of rushing plays per game.

**opponents\_rush\_first\_downs**

Returns a `float` of the opponents' average number of first downs from rushing plays per game.

**opponents\_rush\_touchdowns**

Returns a `float` of the opponents' average number of rushing touchdowns scored per game.

**opponents\_rush\_yards**

Returns a `float` of the opponents' average number of yards gained from rushing per game.

**opponents\_rush\_yards\_per\_attempt**

Returns a `float` of the opponents' average number of yards gained per rushing attempt per game.

**opponents\_turnovers**

Returns a `float` of the opponents' average number of turnovers per game.

**opponents\_yards**

Returns a `float` of the opponents' average number of yards gained per game.

**opponents\_yards\_from\_penalties**

Returns a `float` of the opponents' average number of yards gained from an opponent's penalties per game.

**opponents\_yards\_per\_play**

Returns a `float` of the opponents' average number of yards gained per play.

**pass\_attempts**

Returns a `float` of the average number of passes that are attempted per game.

**pass\_completion\_percentage**

Returns a `float` of the percentage of completed passes per game. Percentage ranges from 0-100.

**pass\_completions**

Returns a `float` of the average number of completed passes per game.

**pass\_first\_downs**

Returns a `float` of the average number of first downs from passing plays per game.

**pass\_touchdowns**

Returns a `float` of the average number of passing touchdowns scored per game.

**pass\_yards**

Returns a `float` of the average number of yards gained from passing per game.

**penalties**

Returns a `float` of the average number of penalties conceded per game.

**plays**

Returns a `float` of the average number of offensive plays per game.

**points\_against\_per\_game**

Returns a `float` of the average number of points conceded per game.

**points\_per\_game**

Returns a `float` of the average number of points scored by the team per game.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**rush\_attempts**

Returns a `float` of the average number of rushing plays per game.

**rush\_first\_downs**

Returns a `float` of the average number of first downs from rushing plays per game.

**rush\_touchdowns**

Returns a `float` of the average number of rushing touchdowns scored per game.

**rush\_yards**

Returns a `float` of the average number of yards gained from rushing per game.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing attempt per game.

**schedule**

Returns an instance of the Schedule class containing the team's complete schedule for the season.



**simple\_rating\_system**

Returns a `float` of the team's relative strength based on the average margin of victory and the strength of schedule. An average team is denoted with 0.0 while a negative score indicates a comparatively weak team.

**strength\_of\_schedule**

Returns a `float` of the team's strength of schedule based on the number of points above or below average. An average difficulty schedule is denoted with 0.0 while a negative score indicates a comparatively easy schedule.

**turnovers**

Returns a `float` of the average number of turnovers per game.

**win\_percentage**

Returns a `float` of the percentage of wins divided by the number of games played during the season. Percentage ranges from 0-1.

**wins**

Returns an `int` of the total number of games the team won during the season.

**yards**

Returns a `float` of the average number of yards gained per game.

**yards\_from\_penalties**

Returns a `float` of the average number of yards gained from an opponent's penalties per game.

**yards\_per\_play**

Returns a `float` of the average number of yards gained per play.

**class** `sportsreference.ncaaf.teams.Teams` (*year=None*)

Bases: `object`

A list of all NCAA Men's Football teams and their stats in a given year.

Finds and retrieves a list of all NCAA Men's Football teams from [www.sports-reference.com](http://www.sports-reference.com) and creates a `Team` instance for every team that participated in the league in a given year. The `Team` class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** `year` (*string optional*) – The requested year to pull stats from.

**dataframes**

Returns a `pandas DataFrame` where each row is a representation of the `Team` class. Rows are indexed by the team abbreviation.

### 1.6.1.5 NFL Package

The NFL package offers multiple modules which can be used to retrieve information and statistics for the National Football League, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of points scored to the number of passing yards, to the number of yards lost from sacks and much more. The Boxscore can be easily queried by passing a boxscore's URI on [sports-reference.com](http://sports-reference.com) which can be retrieved from the `Schedule` class (see `Schedule` module below for more information on retrieving game-specific information).

```
from sportsreference.nfl.boxscore import Boxscore

game_data = Boxscore('201802040nwe')
print(game_data.home_points)  # Prints 33
print(game_data.away_points)  # Prints 41
df = game_data.dataframe  # Returns a Pandas DataFrame of game metrics
```

The Boxscore module also contains a Boxscores class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.

```
from sportsreference.nfl.boxscore import Boxscores

games_today = Boxscores(1, 2017)
# Prints a dictionary of all matchups for week 1 of 2017
print(games_today.games)
```

The Boxscores class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```
from sportsreference.nfl.boxscore import Boxscores

# Pulls all games from weeks 7 and 8 in 2017
games = Boxscores(7, 2017, 8)
# Prints a dictionary of all games from weeks 7 and 8 in 2017
print(games.games)
```

**class** sportsreference.nfl.boxscore.Boxscore(*uri*)  
Bases: object

Detailed information about the final statistics for a game.

Stores all relevant information for a game such as the date, time, location, result, and more advanced metrics such as the number of yards from sacks, a team's passing completion, rushing touchdowns and much more.

**Parameters** *uri* (*string*) – The relative link to the boxscore HTML page, such as '201802040nwe'.

**attendance**

Returns an *int* of the game's listed attendance.

**away\_abbreviation**

Returns a *string* of the away team's abbreviation, such as 'NWE'.

**away\_first\_downs**

Returns an *int* of the number of first downs the away team gained.

**away\_fourth\_down\_attempts**

Returns an *int* of the number of fourth down plays the away team attempted to convert.

**away\_fourth\_down\_conversions**

Returns an *int* of the number of fourth down plays the away team successfully converted.

**away\_fumbles**

Returns an *int* of the number of times the away team fumbled the ball.

**away\_fumbles\_lost**

Returns an *int* of the number of times the away team turned the ball over as the result of a fumble.

**away\_interceptions**

Returns an `int` of the number of interceptions the away team threw.

**away\_net\_pass\_yards**

Returns an `int` of the net pass yards gained by the away team.

**away\_pass\_attempts**

Returns an `int` of the number of passes that were thrown by the away team.

**away\_pass\_completions**

Returns an `int` of the number of completed passes the away team made.

**away\_pass\_touchdowns**

Returns an `int` of the number of passing touchdowns the away team scored.

**away\_pass\_yards**

Returns an `int` of the number of passing yards the away team gained.

**away\_penalties**

Returns an `int` of the number of penalties called on the away team.

**away\_players**

Returns a list of `BoxscorePlayer` class instances for each player on the away team.

**away\_points**

Returns an `int` of the number of points the away team scored.

**away\_rush\_attempts**

Returns an `int` of the number of rushing plays the away team made.

**away\_rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the away team scored.

**away\_rush\_yards**

Returns an `int` of the number of rushing yards the away team gained.

**away\_third\_down\_attempts**

Returns an `int` of the number of third down plays the away team attempted to convert.

**away\_third\_down\_conversions**

Returns an `int` of the number of third down plays the away team successfully converted.

**away\_time\_of\_possession**

Returns a `string` of the amount of time the home team had possession of the football in the format 'MM:SS'.

**away\_times\_sacked**

Returns an `int` of the number of times the away team was sacked.

**away\_total\_yards**

Returns an `int` of the total number of yards the away team gained.

**away\_turnovers**

Returns an `int` of the number of times the away team turned the ball over.

**away\_yards\_from\_penalties**

Returns an `int` of the number of yards gifted as a result of penalties called on the away team.

**away\_yards\_lost\_from\_sacks**

Returns an `int` of the number of yards the away team lost as the result of a sack.

**dataframe**

Returns a pandas `DataFrame` containing all other class properties and values. The index for the `DataFrame` is the string URI that is used to instantiate the class, such as '201802040nwe'.

**date**

Returns a `string` of the date the game took place.

**duration**

MM'.

**Type** Returns a `string` of the game's duration in the format 'H

**home\_abbreviation**

Returns a `string` of the home team's abbreviation, such as 'KAN'.

**home\_first\_downs**

Returns an `int` of the number of first downs the home team gained.

**home\_fourth\_down\_attempts**

Returns an `int` of the number of fourth down plays the home team attempted to convert.

**home\_fourth\_down\_conversions**

Returns an `int` of the number of fourth down plays the home team successfully converted.

**home\_fumbles**

Returns an `int` of the number of times the home team fumbled the ball.

**home\_fumbles\_lost**

Returns an `int` of the number of times the home team turned the ball over as the result of a fumble.

**home\_interceptions**

Returns an `int` of the number of interceptions the home team threw.

**home\_net\_pass\_yards**

Returns an `int` of the net pass yards gained by the home team.

**home\_pass\_attempts**

Returns an `int` of the number of passes that were thrown by the home team.

**home\_pass\_completions**

Returns an `int` of the number of completed passes the home team made.

**home\_pass\_touchdowns**

Returns an `int` of the number of passing touchdowns the home team scored.

**home\_pass\_yards**

Returns an `int` of the number of passing yards the home team gained.

**home\_penalties**

Returns an `int` of the number of penalties called on the home team.

**home\_players**

Returns a `list` of `BoxscorePlayer` class instances for each player on the home team.

**home\_points**

Returns an `int` of the number of points the home team scored.

**home\_rush\_attempts**

Returns an `int` of the number of rushing plays the home team made.

**home\_rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the home team scored.

**home\_rush\_yards**

Returns an `int` of the number of rushing yards the home team gained.

**home\_third\_down\_attempts**

Returns an `int` of the number of third down plays the home team attempted to convert.

**home\_third\_down\_conversions**

Returns an `int` of the number of third down plays the home team successfully converted.

**home\_time\_of\_possession**

Returns a `string` of the amount of time the home team had possession of the football in the format 'MM:SS'.

**home\_times\_sacked**

Returns an `int` of the number of times the home team was sacked.

**home\_total\_yards**

Returns an `int` of the total number of yards the home team gained.

**home\_turnovers**

Returns an `int` of the number of times the home team turned the ball over.

**home\_yards\_from\_penalties**

Returns an `int` of the number of yards gifted as a result of penalties called on the home team.

**home\_yards\_lost\_from\_sacks**

Returns an `int` of the number of yards the home team lost as the result of a sack.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'KAN' for the Kansas City Chiefs.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Kansas City Chiefs'.

**stadium**

Returns a `string` of the name of the stadium where the game was played.

**time**

Returns a `string` of the time the game started.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'NWE' for the New England Patriots.

**winning\_name**

Returns a `string` of the winning team's name, such as 'New England Patriots'.

**class** sportsreference.nfl.boxscore.BoxscorePlayer(*player\_id*, *player\_name*,  
*player\_data*)

Bases: *sportsreference.nfl.player.AbstractPlayer*

Get player stats for an individual game.

Given a player ID, such as 'BreeDr01' for Drew Brees, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

**Parameters**

- **player\_id** (*string*) – A player's ID according to pro-football-reference.com, such as 'BreeDr00' for Drew Brees. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.htm' in the URL. In general,

the ID is in the format 'LlIlFfNN' where 'Llll' are the first 4 letters in the player's last name with the first letter capitalized, 'Ff' are the first 2 letters in the player's first name where the first letter is capitalized, and 'NN' is a number starting at '00' for the first time that player ID has been used and increments by 1 for every successive player.

- **player\_name** (*string*) – A string representing the player's first and last name, such as 'David Blough'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**average\_kickoff\_return\_yards**

Returns a `float` of the average number of yards the player returned a kickoff for.

**combined\_tackles**

Returns an `int` of the number of solo and assisted tackles the player made.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values for the specified game.

**fumbles\_lost**

Returns an `int` of the number of times the player fumbled the ball and the opponent recovered the ball.

**quarterback\_hits**

Returns an `int` of the number of times the player hit the quarterback.

**solo\_tackles**

Returns an `int` of the number of solo tackles the player made during the game.

**tackles\_for\_loss**

Returns an `int` of the number of times the player tackles an opponent for a loss on the play.

**yards\_lost\_from\_sacks**

Returns an `int` of the total number of yards the player lost after being sacked by the opponent.

**class** `sportsreference.nfl.boxscore.Boxscores` (*week, year, end\_week=None*)

Bases: `object`

Search for NFL games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

**Parameters**

- **week** (*int*) – The week number to pull games from.
- **year** (*int*) – The 4-digit year to pull games from.
- **end\_week** (*int (optional)*) – Optionally specify an end week to iterate until. All boxscores starting from the week specified in the 'week' parameter up to and including the boxscores specified in the 'end\_week' parameter will be pulled. If left empty, or if 'end\_week' is prior to 'week', only the games from the day specified in the 'date' parameter will be saved.

**games**

Returns a `dictionary` object representing all of the games played on the requested day. Dictionary is in the following format:

```
{'week' : [ # 'week' is the string week in format 'W-YYYY'
    {
        'home_name': Name of the home team, such as 'Kansas City
            Chiefs' (`str`),
        'home_abbr': Abbreviation for the home team, such as 'KAN'
            (`str`),
        'away_name': Name of the away team, such as 'Houston
            Texans' (`str`),
        'away_abbr': Abbreviation for the away team, such as 'HTX'
            (`str`),
        'boxscore': String representing the boxscore URI, such as
            'SLN/SLN201807280' (`str`),
        'winning_name': Full name of the winning team, such as
            'Kansas City Chiefs' (`str`),
        'winning_abbr': Abbreviation for the winning team, such as
            'KAN' (`str`),
        'losing_name': Full name of the losing team, such as
            'Houston Texans' (`str`),
        'losing_abbr': Abbreviation for the losing team, such as
            'HTX' (`str`),
        'home_score': Integer score for the home team (`int`),
        'away_score': Integer score for the away team (`int`)
    },
    { ... },
    ...
  ]
}
```

If no games were played on ‘week’, the list for [‘week’] will be empty.

sportsreference.nfl.boxscore.nfl\_int\_property\_sub\_index (*func*)

## Player

The Player module contains an abstract base class that can be inherited by both the BoxscorePlayer and Player classes in the Boxscore and Roster modules, respectively. All of the properties that appear in the AbstractPlayer class can be read from either of the two child classes mentioned above.

**class** sportsreference.nfl.player.**AbstractPlayer** (*player\_id, player\_name, player\_data*)

Bases: object

Get player information and stats for all seasons.

Given a player ID, such as ‘BreeDr00’ for Drew Brees, capture all relevant stats and information like name, team, height/weight, career starts, single season passing yards, sacks, and much more.

By default, the class instance will return the player’s career stats, but single-season stats can be found by calling the instance with the requested season as denoted on pro-football-reference.com.

### Parameters

- **player\_id** (*string*) – A player’s ID according to pro-football-reference.com, such as ‘BreeDr00’ for Drew Brees. The player ID can be found by navigating to the player’s stats page and getting the string between the final slash and the ‘.htm’ in the URL. In general, the ID is in the format ‘LIIIFNN’ where ‘LIII’ are the first 4 letters in the player’s last name with the first letter capitalized, ‘Ff’ are the first 2 letters in the player’s first name where the first letter is capitalized, and ‘NN’ is a number starting at ‘00’ for the first time that player ID has been used and increments by 1 for every successive player.

- **player\_name** (*string*) – A string representing the player’s first and last name, such as ‘Drew Brees’.
- **player\_data** (*string*) – A string representation of the player’s HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

**assists\_on\_tackles**

Returns an `int` of the number of assist the player made on tackles.

**attempted\_passes**

Returns an `int` of the number of passes the player attempted.

**completed\_passes**

Returns an `int` of the number of completed passes the player threw.

**extra\_points\_attempted**

Returns an `int` of the number of extra points the player attempted.

**extra\_points\_made**

Returns an `int` of the number of extra points the player made.

**field\_goals\_attempted**

Returns an `int` of the total number of field goals the player attempted from any distance.

**field\_goals\_made**

Returns an `int` of the total number of field goals the player made from any distance.

**fumbles**

Returns an `int` of the number of times the player fumbled the ball.

**fumbles\_forced**

Returns an `int` of the number of times the player forced a fumble.

**fumbles\_recovered**

Returns an `int` of the number of fumbles the player has recovered.

**fumbles\_recovered\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after recovering a fumble.

**interceptions**

Returns an `int` of the number of times the player intercepted a pass.

**interceptions\_returned\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after intercepting a pass. Commonly referred to as a ‘Pick-6’.

**interceptions\_thrown**

Returns an `int` of the number of interceptions the player has thrown.

**kickoff\_return\_touchdown**

Returns an `int` of the number of kickoffs the player returned for a touchdown.

**kickoff\_return\_yards**

Returns an `int` of the amount of yards the player gained while returning a kickoff.

**kickoff\_returns**

Returns an `int` of the number of kickoffs the player returned.

**longest\_interception\_return**

Returns an `int` of the most yards the player has returned after intercepting a pass.

**longest\_kickoff\_return**

Returns an `int` of the highest number of yards the player has gained while returning a kickoff.



**longest\_pass**

Returns an `int` of the longest completed pass the player threw.

**longest\_punt**

Returns an `int` of the longest punt the player has kicked.

**longest\_punt\_return**

Returns an `int` of the highest number of yards the player has gained while returning a punt.

**longest\_reception**

Returns an `int` of the highest number of yards the player gained as a result of a single reception.

**longest\_rush**

Returns an `int` of the highest number of yards the player gained during a single rushing attempt.

**name**

Returns a `string` of the player's name, such as 'Drew Brees'.

**passes\_defended**

Returns an `int` of the number of passes the player has defended as a defensive player.

**passing\_touchdowns**

Returns an `int` of the number of touchdowns passes the player has thrown.

**passing\_yards**

Returns an `int` of the number of yards receivers have gained as a result of the player's passes.

**player\_id**

Returns a `string` of the player's ID on pro-football-reference, such as 'BreeDr00' for Drew Brees.

**punt\_return\_touchdown**

Returns an `int` of the number of punts the player returned for a touchdown.

**punt\_return\_yards**

Returns an `int` of the amount of yards the player gained while returning a punt.

**punt\_returns**

Returns an `int` of the number of times a player returned a punt.

**punts**

Returns an `int` of the number of times the player punted the ball.

**quarterback\_rating**

Returns a `float` of the player's quarterback rating.

**receiving\_touchdowns**

Returns an `int` of the number of touchdowns the player scored after receiving a pass.

**receiving\_yards**

Returns an `int` of the number of receiving yards the player gained.

**receptions**

Returns an `int` of the number of receptions the player made.

**rush\_attempts**

Returns an `int` of the number of rushing plays the player attempted.

**rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the player scored.

**rush\_yards**

Returns an `int` of the number of rushing yards the player gained.

**sacks**

Returns a float of the number of times the player sacked a quarterback.

**times\_pass\_target**

Returns an int of the number of times the player was the target of a pass.

**times\_sacked**

Returns an int of the number of times the player was sacked as a quarterback.

**total\_punt\_yards**

Returns an int of the total number of yards the player has punted the ball.

**yards\_per\_punt**

Returns a float of the average distance the player punts the ball.

**yards\_per\_punt\_return**

Returns a float of the average number of yards the player returned per punt.

**yards\_recovered\_from\_fumble**

Returns an int of the number of yards the player gained after recovering a fumble.

**yards\_returned\_from\_interception**

Returns an int of the number of yards the player returned after intercepting a pass.

## Roster

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the `Player` class which has detailed information ranging from career touchdowns to single-season stats and player height, weight, and nationality. The following is an example on collecting career information for Drew Brees.

```
from sportsreference.nfl.roster import Player

brees = Player('BreeDr00')
print(brees.name) # Prints 'Drew Brees'
print(brees.passing_yards) # Prints Brees' career passing yards
# Prints a Pandas DataFrame of all relevant stats per season for Brees
print(brees.dataframe)
```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific team, call the class instance with the season string. All future property requests will return the season-specific stats.

```
from sportsreference.nfl.roster import Player

brees = Player('BreeDr00') # Currently pulling career stats
print(brees.passing_yards) # Prints Brees' CAREER passing yards total
# Prints Brees' passing yards total only for the 2017 season
print(brees('2017').passing_yards)
# Prints Brees' passing touchdowns for the 2017 season only
print(brees.passing_touchdowns)
```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.nfl.roster import Player

brees = Player('BreeDr00') # Currently pulling career stats
# Prints Brees' passing yards total only for the 2017 season
```

(continues on next page)

(continued from previous page)

```
print(brees('2017').passing_yards)
print(brees('Career').passing_yards)  # Prints Brees' career passing yards
```

In addition, the Roster module also contains the `Roster` class which can be used to pull all players on a team's roster during a given season and creates instances of the `Player` class for each team member and adds them to a list to be easily queried.

```
from sportsreference.nfl.roster import Roster

saints = Roster('NOR')
for player in saints.players:
    # Prints the name of all players who played for the New Orleans Saints
    # in the most recent season.
    print(player.name)
```

**class** `sportsreference.nfl.roster.Player` (*player\_id*)  
 Bases: `sportsreference.nfl.player.AbstractPlayer`

Get player information and stats for all seasons.

Given a player ID, such as 'BreeDr00' for Drew Brees, capture all relevant stats and information like name, team, height/weight, career starts, single season passing yards, sacks, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on [pro-football-reference.com](http://pro-football-reference.com).

**Parameters** `player_id` (*string*) – A player's ID according to [pro-football-reference.com](http://pro-football-reference.com), such as 'BreeDr00' for Drew Brees. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.htm' in the URL. In general, the ID is in the format 'LIIIFfNN' where 'LIII' are the first 4 letters in the player's last name with the first letter capitalized, 'Ff' are the first 2 letters in the player's first name where the first letter is capitalized, and 'NN' is a number starting at '00' for the first time that player ID has been used and increments by 1 for every successive player.

**adjusted\_net\_yards\_per\_attempt\_index**

Returns an `int` comparing players by the net average adjusted yards gained per attempt where 100 denotes an average player in this category and higher numbers are better.

**adjusted\_net\_yards\_per\_pass\_attempt**

Returns a `float` of the adjusted net yards gained per pass attempt, equal to  $(\text{pass\_yards} - \text{sack\_yards} + (20 * \text{pass\_touchdowns}) - (45 * \text{interceptions})) / (\text{pass\_attempts} + \text{times\_sacked})$ .

**adjusted\_yards\_per\_attempt**

Returns a `float` of the adjusted number of yards gained per passing attempt, equal to  $(\text{yards} + 20 * \text{pass\_touchdowns} - 45 * \text{interceptions}) / \text{pass\_attempts}$ .

**adjusted\_yards\_per\_attempt\_index**

Returns an `int` comparing players by the average adjusted yards gained per attempt where 100 denotes an average player in this category and higher numbers are better.

**all\_purpose\_yards**

Returns an `int` of the number of all-purpose yards the player has gained from receptions, rushes, and kickoff and punt returns.

**approximate\_value**

Returns an `int` of the player's approximate value which is a singular number used to compare players across seasons and positions, but is only intended to be a rough estimate.

**assists\_on\_tackles**

Returns an `int` of the number of assist the player made on tackles.

**attempted\_passes**

Returns an `int` of the number of passes the player attempted.

**birth\_date**

Returns a `datetime` object of the day and year the player was born.

**blocked\_punts**

Returns an `int` of the number of the player's punts that have been blocked.

**catch\_percentage**

Returns a `float` of the percentage of passes the player caught while being the target of a pass. Percentage ranges from 0-100.

**completed\_passes**

Returns an `int` of the number of completed passes the player threw.

**completion\_percentage\_index**

Returns an `int` comparing players by their passing completion percentage where 100 denotes an average player in this category and higher numbers are better.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values where each index is a different season plus the career stats.

**espn\_qbr**

Returns a `float` of the player's Total Quarterback Rating according to ESPN.

**extra\_point\_percentage**

Returns a `float` of the percentage of extra points the player made. Percentage ranges from 0-100.

**extra\_points\_attempted**

Returns an `int` of the number of extra points the player attempted.

**extra\_points\_made**

Returns an `int` of the number of extra points the player made.

**field\_goal\_percentage**

Returns a `float` of the percentage of field goals the player made. Percentage ranges from 0-100.

**field\_goals\_attempted**

Returns an `int` of the total number of field goals the player attempted from any distance.

**field\_goals\_made**

Returns an `int` of the total number of field goals the player made from any distance.

**fifty\_plus\_yard\_field\_goal\_attempts**

Returns an `int` of the number of field goals the player attempted from fifty or more yards out.

**fifty\_plus\_yard\_field\_goals\_made**

Returns an `int` of the number of field goals the player made from fifty or more yards out.

**fourth\_quarter\_comebacks**

Returns an `int` of the number of times the player has lead a team to victory or a tie as a quarterback while the team trailed at the beginning of the fourth quarter by scoring at the end of a drive.

**fourty\_to\_fourty\_nine\_yard\_field\_goal\_attempts**

Returns an `int` of the number of field goals the player attempted from fourty to fourty-nine yards out.

**fourty\_to\_fourty\_nine\_yard\_field\_goals\_made**

Returns an `int` of the number of field goals the player made from fourty to fourty-nine yards out.

**fumbles**

Returns an `int` of the number of times the player fumbled the ball.

**fumbles\_forced**

Returns an `int` of the number of times the player forced a fumble.

**fumbles\_recovered**

Returns an `int` of the number of fumbles the player has recovered.

**fumbles\_recovered\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after recovering a fumble.

**game\_winning\_drives**

Returns an `int` of the number of times the player has lead a drive that resulted in a score in the fourth quarter while the team was trailing.

**games**

Returns an `int` of the number of games the player participated in.

**games\_started**

Returns an `int` of the number of games the player started.

**height**

Returns a `string` of the player's height in the format "feet-inches".

**interception\_percentage**

Returns a `float` of the percentage of passes the player throws that are intercepted. Percentage ranges from 0-100.

**interception\_percentage\_index**

Returns an `int` comparing players by the percentage of their passes that are intercepted where 100 denotes an average player in this category and higher numbers are better.

**interceptions**

Returns an `int` of the number of times the player intercepted a pass.

**interceptions\_returned\_for\_touchdown**

Returns an `int` of the number of touchdowns the player has scored after intercepting a pass. Commonly referred to as a 'Pick-6'.

**interceptions\_thrown**

Returns an `int` of the number of interceptions the player has thrown.

**kickoff\_return\_touchdown**

Returns an `int` of the number of kickoffs the player returned for a touchdown.

**kickoff\_return\_yards**

Returns an `int` of the amount of yards the player gained while returning a kickoff.

**kickoff\_returns**

Returns an `int` of the number of kickoffs the player returned.

**less\_than\_nineteen\_yards\_field\_goal\_attempts**

Returns an `int` of the number of field goals the player attempted from nineteen or fewer yards out.

**less\_than\_nineteen\_yards\_field\_goals\_made**

Returns an `int` of the number of field goals the player made from nineteen or fewer yards out.

**longest\_field\_goal\_made**

Returns an `int` of the longest field goal the player made.

**longest\_interception\_return**

Returns an `int` of the most yards the player has returned after intercepting a pass.

**longest\_kickoff\_return**

Returns an `int` of the highest number of yards the player has gained while returning a kickoff.

**longest\_pass**

Returns an `int` of the longest completed pass the player threw.

**longest\_punt**

Returns an `int` of the longest punt the player has kicked.

**longest\_punt\_return**

Returns an `int` of the highest number of yards the player has gained while returning a punt.

**longest\_reception**

Returns an `int` of the highest number of yards the player gained as a result of a single reception.

**longest\_rush**

Returns an `int` of the highest number of yards the player gained during a single rushing attempt.

**net\_yards\_per\_attempt\_index**

Returns an `int` comparing players by the net average yards gained per attempt where 100 denotes an average player in this category and higher numbers are better.

**net\_yards\_per\_pass\_attempt**

Returns a `float` of the net yards gained per pass attempt, equal to  $(\text{pass\_yards} - \text{sack\_yards}) / (\text{pass\_attempts} + \text{times\_sacked})$ .

**passer\_rating\_index**

Returns an `int` comparing players by their quarterback rating where 100 denotes an average player in this category and higher numbers are better.

**passes\_defended**

Returns an `int` of the number of passes the player has defended as a defensive player.

**passing\_completion**

Returns a `float` of the percentage of passes that were caught by a receiver. Percentage ranges from 0-100.

**passing\_touchdown\_percentage**

Returns a `float` of the percentage of total passes that are touchdowns. Percentage ranges from 0-100.

**passing\_touchdowns**

Returns an `int` of the number of touchdowns passes the player has thrown.

**passing\_yards**

Returns an `int` of the number of yards receivers have gained as a result of the player's passes.

**passing\_yards\_per\_attempt**

Returns a `float` of the number of yards gained per passing attempt.

**position**

Returns a `string` of the player's primary position.

**punt\_return\_touchdown**

Returns an `int` of the number of punts the player returned for a touchdown.

**punt\_return\_yards**

Returns an `int` of the amount of yards the player gained while returning a punt.

**punt\_returns**

Returns an `int` of the number of times a player returned a punt.

**punts**

Returns an `int` of the number of times the player punted the ball.

**qb\_record**

Returns a `string` of the player's quarterback record as a starter in the format 'W-L-T'.

**quarterback\_rating**

Returns a `float` of the player's quarterback rating.

**receiving\_touchdowns**

Returns an `int` of the number of touchdowns the player scored after receiving a pass.

**receiving\_yards**

Returns an `int` of the number of receiving yards the player gained.

**receiving\_yards\_per\_game**

Returns a `float` of the average number of receiving yards the player gains per game.

**receiving\_yards\_per\_reception**

Returns a `float` of the average number of yards the player gained per reception.

**receptions**

Returns an `int` of the number of receptions the player made.

**receptions\_per\_game**

Returns a `float` of the average number of receptions the player makes per game.

**rush\_attempts**

Returns an `int` of the number of rushing plays the player attempted.

**rush\_attempts\_per\_game**

Returns a `float` of the average number of rushing attempts the player made per game.

**rush\_touchdowns**

Returns an `int` of the number of rushing touchdowns the player scored.

**rush\_yards**

Returns an `int` of the number of rushing yards the player gained.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing attempt.

**rush\_yards\_per\_game**

Returns a `float` of the average number of rushing yards gained per game.

**rushing\_and\_receiving\_touchdowns**

Returns an `int` of the combined number of rushing and receiving touchdowns the player scored.

**sack\_percentage**

Returns a `float` of the percentage of times sacked during a passing attempt, equal to `times_sacked / (pass_attempts + times_sacked)`. Percentage ranges from 0-100.

**sack\_percentage\_index**

Returns an `int` comparing players by the percentage of plays that end in the player being sacked where 100 denotes an average player in this category and higher numbers are better.

**sacks**

Returns a `float` of the number of times the player sacked a quarterback.

**safeties**

Returns an `int` of the number of safeties the player has scored.

**season**

Returns a `string` of the season in the format 'YYYY', such as '2017'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**tackles**

Returns an `int` of the number of tackles the player made.

**team\_abbreviation**

Returns a `string` of the team's abbreviation, such as 'NOR' for the New Orleans Saints.

**thirty\_to\_thirty\_nine\_yard\_field\_goal\_attempts**

Returns an `int` of the number of field goals the player attempted from thirty to thirty-nine yards out.

**thirty\_to\_thirty\_nine\_yard\_field\_goals\_made**

Returns an `int` of the number of field goals the player made from thirty to thirty-nine yards out.

**times\_pass\_target**

Returns an `int` of the number of times the player was the target of a pass.

**times\_sacked**

Returns an `int` of the number of times the player was sacked as a quarterback.

**total\_punt\_yards**

Returns an `int` of the total number of yards the player has punted the ball.

**touchdown\_percentage\_index**

Returns an `int` comparing players by the percentage of their passes that result in a touchdown where 100 denotes an average player in this category and higher numbers are better.

**touches**

Returns an `int` of the combined number of rushing attempts and receptions the player had.

**twenty\_to\_twenty\_nine\_yard\_field\_goal\_attempts**

Returns an `int` of the number of field goals the player attempted from twenty to twenty-nine yards out.

**twenty\_to\_twenty\_nine\_yard\_field\_goals\_made**

Returns an `int` of the number of field goals the player made from twenty to twenty-nine yards out.

**weight**

Returns an `int` of the player's weight in pounds.

**yards\_from\_scrimmage**

Returns an `int` of the total number of yards gained from scrimmage for both rushing and receiving.

**yards\_lost\_to\_sacks**

Returns an `int` of the number of yards lost as a result of sacks.

**yards\_per\_attempt\_index**

Returns an `int` comparing players by the average number of yards gained per attempt where 100 denotes an average player in this category and higher numbers are better.

**yards\_per\_completed\_pass**

Returns a `float` of the number of yards gained per completed pass.

**yards\_per\_game\_played**

Returns a `float` of the number of passing yards gained per game.

**yards\_per\_kickoff\_return**

Returns a `float` of the average number of yards the player returned per kickoff.

**yards\_per\_punt\_return**

Returns a `float` of the average number of yards the player returned per punt.

**yards\_per\_touch**

Returns a `float` of the average number of yards gained per rushing attempt and/or reception.

**yards\_recovered\_from\_fumble**

Returns an `int` of the number of yards the player gained after recovering a fumble.



**yards\_returned\_from\_interception**

Returns an int of the number of yards the player returned after intercepting a pass.

**class** sportsreference.nfl.roster.**Roster**(*team*, *year=None*, *slim=False*)

Bases: object

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the Player class for each player, containing a detailed list of the player's statistics and information.

**Parameters**

- **team**(*string*) – The team's abbreviation, such as 'NOR' for the New Orleans Saints.
- **year**(*string (optional)*) – The 4-digit year to pull the roster from, such as '2017'. If left blank, defaults to the most recent season.
- **slim**(*boolean (optional)*) – Set to True to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to False.

**players**

Returns a list of player instances for each player on the requested team's roster if the `slim` property is False when calling the Roster class. If the `slim` property is True, returns a dictionary where each key is a string of the player's ID and each value is the player's first and last name as listed on the roster page.

**Schedule**

The Schedule module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.nfl.schedule import Schedule

houston_schedule = Schedule('HTX')
for game in houston_schedule:
    print(game.date)  # Prints the date the game was played
    print(game.result) # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** sportsreference.nfl.schedule.**Game**(*game\_data*, *game\_type*, *year*)

Bases: object

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

**Parameters**

- **game\_data**(*string*) – The row containing the specified game's information.
- **game\_type**(*string*) – A constant to denote whether a game took place in the regular season or in the playoffs.
- **year**(*string*) – The year as a 4-digit string. Note that this is the year that the bulk of the season took place. For example the Super Bowl for the 2017 season took place in early

February 2018, but 2017 should be passed as that was the year the bulk of the season was played in.

**boxscore**

Returns an instance of the Boxscore class containing more detailed stats on the game.

**boxscore\_index**

Returns a `string` of the URI for a boxscore which can be used to access or index a game.

**dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame representing the Boxscore class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter ‘dataframe’ property. The index for the DataFrame is the boxscore string.

**date**

Returns a `string` of the month and day the game was played, such as ‘September 7’.

**datetime**

Returns a datetime object representing the date the game was played.

**day**

Returns a `string` of the day of the week the game was played as a 3-letter abbreviation, such as ‘Sun’ for Sunday.

**extra\_points\_attempted**

Returns an `int` of the number of times the team attempted to convert an extra point after scoring a touchdown.

**extra\_points\_made**

Returns an `int` of the number of extra points the team successfully converted after scoring a touchdown.

**field\_goals\_attempted**

Returns an `int` of the total number of times the team attempted a field goal.

**field\_goals\_made**

Returns an `int` of the total number of field goals the team scored.

**fourth\_down\_attempts**

Returns an `int` of the total number of fourth downs the team attempted to convert.

**fourth\_down\_conversions**

Returns an `int` of the number of fourth downs the team successfully converted.

**interceptions**

Returns an `int` of the number of interceptions the team threw.

**location**

Returns a `string` constant indicating whether the game was played at home, away, or a neutral site, such as the Super Bowl.

**opponent\_abbr**

Returns a `string` of the opponent’s 3-letter abbreviation, such as ‘NWE’ for the New England Patriots.

**opponent\_name**

Returns a `string` of the opponent’s full name, such as the ‘New England Patriots’.

**overtime**

Returns a `boolean` value that evaluates to True if the game went to overtime and False if it ended in regulation.

**pass\_attempts**

Returns an `int` of the number of passes the team attempted during the game.

**pass\_completion\_rate**

Returns a `float` of the percentage of passes that were completed by the team. Percentage ranges from 0-100.

**pass\_completions**

Returns an `int` of the number of completed passes by the team.

**pass\_touchdowns**

Returns an `int` of the number of touchdowns the team scored as a result of passing plays.

**pass\_yards**

Returns an `int` of the number of yards the team gained as a result of passing plays.

**pass\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per passing play.

**points\_allowed**

Returns an `int` of the number of points allowed by the team.

**points\_scored**

Returns an `int` of the number of points scored by the team.

**punt\_yards**

Returns an `int` of the total number of yards the team punted the ball.

**punts**

Returns an `int` of the number of times the team punted the ball.

**quarterback\_rating**

Returns a `float` of the quarterback's rating for the game.

**result**

Returns a `string` constant indicating whether the team won or lost the game.

**rush\_attempts**

Returns an `int` of the total number of times the team attempted a rushing play.

**rush\_touchdowns**

Returns an `int` of the number of touchdowns the team scored as a result of rushing plays.

**rush\_yards**

Returns an `int` of the total number of yards the team gain as a result of rushing plays.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing play.

**third\_down\_attempts**

Returns an `int` of the total number of third downs the team attempted to convert.

**third\_down\_conversions**

Returns an `int` of the number of third downs the team successfully converted.

**time\_of\_possession**

Returns a `string` of the total time the team spent with the ball. Time is in the format 'MM:SS'.

**times\_sacked**

Returns an `int` of the number of times the quarterback was sacked by the opponent.

**type**

Returns a `string` constant indicating whether the game is a regular season or playoff matchup.

**week**

Returns an `int` of the week number in the season, such as 1 for the first week of the regular season.

**yards\_lost\_from\_sacks**

Returns an `int` of the total number of yards lost as a result of a sack.

**class** `sportsreference.nfl.schedule.Schedule` (*abbreviation*, *year=None*)

Bases: `object`

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

**Parameters**

- **abbreviation** (*string*) – A team's short name, such as 'NWE' for the New England Patriots.
- **year** (*string* (*optional*)) – The requested year to pull stats from.

**dataframe**

Returns a pandas DataFrame where each row is a representation of the Game class. Rows are indexed by the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame where each row is a representation of the Boxscore class for every game in the schedule. Rows are indexed by the boxscore string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The Teams module exposes information for all MLB teams including the team name and abbreviation, the number of games they won during the season, the average margin of victory, and much more.

```
from sportsreference.nfl.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    # Prints the team's average margin of victory
    print(team.margin_of_victory)
```

Each Team instance contains a link to the Schedule class which enables easy iteration over all games for a particular team. A Pandas DataFrame can also be queried to easily grab all stats for all games.

```
from sportsreference.nfl.teams import Teams

teams = Teams()
for team in teams:
    schedule = team.schedule    # Returns a Schedule instance for each team
    # Returns a Pandas DataFrame of all metrics for all game Boxscores for
    # a season.
    df = team.schedule.dataframe_extended
```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```

from sportsreference.nfl.teams import Teams

for team in Teams():
    roster = team.roster # Gets each team's roster
    for player in roster.players:
        print(player.name) # Prints each players name on the roster

```

**class** sportsreference.nfl.teams.**Team**(*team\_data*, *rank*, *year=None*)

Bases: object

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as rank, name, and abbreviation, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **rank** (*int*) – A team's position in the league based on the number of points they obtained during the season.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a *string* of team's abbreviation, such as 'KAN' for the Kansas City Chiefs.

#### dataframe

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'KAN'.

#### defensive\_simple\_rating\_system

Returns a *float* of the team's defensive strength according to the simple rating system. An average team is denoted with 0.0 and a negative score is a comparatively weaker team.

#### first\_downs

Returns an *int* of the total number of first downs the team achieved during the season.

#### first\_downs\_from\_penalties

Returns an *int* of the total number of first downs conceded as a result of penalties called on the team.

#### fumbles

Returns an *int* of the total number of times the team fumbled the ball during the season.

#### games\_played

Returns an *int* of the number of games played during the season.

#### interceptions

Returns an *int* of the total number of interceptions the team has thrown.

#### losses

Returns an *int* of the number of games the team lost during the season.

#### margin\_of\_victory

Returns a *float* of the average margin of victory per game.

#### name

Returns a *string* of the team's full name, such as 'Kansas City Chiefs'.

**offensive\_simple\_rating\_system**

Returns a `float` of the team's offensive strength according to the simple rating system. An average team is denoted with 0.0 and a negative score is a comparatively weaker team.

**pass\_attempts**

Returns an `int` of the total number of passes that were attempted.

**pass\_completions**

Returns an `int` of the total number of passes that were completed.

**pass\_first\_downs**

Returns an `int` of the number of first downs the team gained from passing plays.

**pass\_net\_yards\_per\_attempt**

Returns a `float` of the net yards gained per passing play including sacks.

**pass\_touchdowns**

Returns an `int` of the total number of touchdowns the team has scored from passing.

**pass\_yards**

Returns an `int` of the total number of yards the team gained from passing.

**penalties**

Returns an `int` of the total number of penalties called on the team during the season.

**percent\_drives\_with\_points**

Returns a `float` of the percentage of drives that result in points for the offense. Percentage ranges from 0-100.

**percent\_drives\_with\_turnovers**

Returns a `float` of the percentage of drives that result in an offensive turnover. Percentage ranges from 0-100.

**plays**

Returns an `int` of the total number of offensive plays the team has made during the season.

**points\_against**

Returns an `int` of the total number of points allowed during the season.

**points\_contributed\_by\_offense**

Returns a `float` of the number of expected points contributed by the offense.

**points\_difference**

Returns an `int` of the difference between the number of points scored and allowed during the season.

**points\_for**

Returns an `int` of the total number of points scored during the season.

**rank**

Returns an `int` of the team's rank based on the number of points they scored during the season.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**rush\_attempts**

Returns an `int` of the total number of rushing plays that were attempted.

**rush\_first\_downs**

Returns an `int` of the total number of first downs gained from rushing plays.

**rush\_touchdowns**

Returns an `int` of the total number of touchdowns from rushing plays.

**rush\_yards**

Returns an `int` of the total number of yards that were gained from rushing plays.

**rush\_yards\_per\_attempt**

Returns a `float` of the average number of yards gained per rushing play.

**schedule**

Returns an instance of the `Schedule` class containing the team's complete schedule for the season.

**simple\_rating\_system**

Returns a `float` of the team's relative strength based on average margin of victory plus strength of schedule. An average team is denoted with 0.0 and a negative score is a comparatively weaker team.

**strength\_of\_schedule**

Returns a `float` of the team's strength of schedule. An average difficulty schedule is denoted with a 0.0 and a negative number is comparatively easier than average.

**turnovers**

Returns an `int` of the total number of turnovers the team committed during the season.

**win\_percentage**

Returns a `float` of the number of wins divided by the number of games played. Percentage ranges from 0-1.

**wins**

Returns an `int` of the number of games the team won during the season.

**yards**

Returns an `int` of the total number of yards the team has gained during the season.

**yards\_from\_penalties**

Returns an `int` of the total number of yards surrendered as a result of penalties called on the team.

**yards\_per\_play**

Returns a `float` of the average number of yards gained per play during the season.

**class** `sportsreference.nfl.teams.Teams` (*year=None*)

Bases: `object`

A list of all NFL teams and their stats in a given year.

Finds and retrieves a list of all NFL teams from [www.pro-football-reference.com](http://www.pro-football-reference.com) and creates a `Team` instance for every team that participated in the league in a given year. The `Team` class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** `year` (*string (optional)*) – The requested year to pull stats from.

**dataframes**

Returns a pandas `DataFrame` where each row is a representation of the `Team` class. Rows are indexed by the team abbreviation.

### 1.6.1.6 NHL Package

The NHL package offers multiple modules which can be used to retrieve information and statistics for the National Hockey League, such as team names, season stats, game schedules, and boxscore metrics.

#### Boxscore

The Boxscore module can be used to grab information from a specific game. Metrics range from number of goals scored to the number of penalty minutes, to the save percentage and much more. The Boxscore can be easily queried by

passing a boxscore's URI on sports-reference.com which can be retrieved from the `Schedule` class (see `Schedule` module below for more information on retrieving game-specific information).

```
from sportsreference.nhl.boxscore import Boxscore

game_data = Boxscore('201806070VEG')
print(game_data.home_goals)  # Prints 3
print(game_data.away_goals)  # Prints 4
df = game_data.dataframe  # Returns a Pandas DataFrame of game metrics
```

The `Boxscore` module also contains a `Boxscores` class which searches for all games played on a particular day and returns a dictionary of matchups between all teams on the requested day. The dictionary includes the names and abbreviations for each matchup as well as the boxscore link if applicable.

```
from datetime import datetime
from sportsreference.nhl.boxscore import Boxscores

games_today = Boxscores(datetime.today())
print(games_today.games)  # Prints a dictionary of all matchups for today
```

The `Boxscores` class also allows the ability to query over a range of dates using a second optional parameter during instantiation of the class. To query a range of dates, enter the start date as the first parameter and the inclusive end date as the second parameter.

```
from datetime import datetime
from sportsreference.nhl.boxscore import Boxscores

# Pulls all games between and including February 4, 2017 and February 5,
# 2017
games = Boxscores(datetime(2017, 2, 4), datetime(2017, 2, 5))
# Prints a dictionary of all results from February 4, 2017 and February 5,
# 2017
print(games.games)
```

**class** `sportsreference.nhl.boxscore.Boxscore` (*uri*)

Bases: `object`

Detailed information about the final statistics for a game.

Stores all relevant information for a game such as the date, time, location, result, and more advanced metrics such as the number of goals scored, the number of points for a player, the amount of power play assists and much more.

**Parameters** *uri* (*string*) – The relative link to the boxscore HTML page, such as '201806070VEG'.

**arena**

Returns a `string` of the name of the ballpark where the game was played.

**attendance**

Returns an `int` of the game's listed attendance.

**away\_assists**

Returns an `int` of the number of assists the away team registered.

**away\_even\_strength\_assists**

Returns an `int` of the number of assists the away team registered while at even strength.

**away\_even\_strength\_goals**

Returns an `int` of the number of goals the away team scored at even strength.



**away\_game\_winning\_goals**

Returns an `int` of the number of game winning goals the away team scored.

**away\_goals**

Returns an `int` of the number of goals the away team scored.

**away\_penalties\_in\_minutes**

Returns an `int` of the length of time the away team spent in the penalty box.

**away\_players**

Returns a `list` of `BoxscorePlayer` class instances for each player on the away team.

**away\_points**

Returns an `int` of the number of points the away team registered.

**away\_power\_play\_assists**

Returns an `int` of the number of assists the away team registered while on a power play.

**away\_power\_play\_goals**

Returns an `int` of the number of goals the away team scored while on a power play.

**away\_save\_percentage**

Returns a `float` of the percentage of shots the away team saved. Percentage ranges from 0-1.

**away\_saves**

Returns an `int` of the number of saves the away team made.

**away\_shooting\_percentage**

Returns a `float` of the away team's shooting percentage. Percentage ranges from 0-100.

**away\_short\_handed\_assists**

Returns an `int` of the number of assists the away team registered while short handed.

**away\_short\_handed\_goals**

Returns an `int` of the number of goals the away team scored while short handed.

**away\_shots\_on\_goal**

Returns an `int` of the number of shots on goal the away team registered.

**away\_shutout**

Returns an `int` denoting whether or not the away team shutout the home team.

**dataframe**

Returns a `pandas DataFrame` containing all other class properties and values. The index for the `DataFrame` is the string URI that is used to instantiate the class, such as '201806070VEG'.

**date**

Returns a `string` of the date the game took place.

**duration**

MM'.

**Type** Returns a `string` of the game's duration in the format 'H

**home\_assists**

Returns an `int` of the number of assists the home team registered.

**home\_even\_strength\_assists**

Returns an `int` of the number of assists the home team registered while at even strength.

**home\_even\_strength\_goals**

Returns an `int` of the number of goals the home team scored at even strength.

**home\_game\_winning\_goals**

Returns an `int` of the number of game winning goals the home team scored.

**home\_goals**

Returns an `int` of the number of goals the home team scored.

**home\_penalties\_in\_minutes**

Returns an `int` of the length of time the home team spent in the penalty box.

**home\_players**

Returns a `list` of `BoxscorePlayer` class instances for each player on the home team.

**home\_points**

Returns an `int` of the number of points the home team registered.

**home\_power\_play\_assists**

Returns an `int` of the number of assists the home team registered while on a power play.

**home\_power\_play\_goals**

Returns an `int` of the number of goals the home team scored while on a power play.

**home\_save\_percentage**

Returns a `float` of the percentage of shots the home team saved. Percentage ranges from 0-1.

**home\_saves**

Returns an `int` of the number of saves the home team made.

**home\_shooting\_percentage**

Returns a `float` of the home team's shooting percentage. Percentage ranges from 0-100.

**home\_short\_handed\_assists**

Returns an `int` of the number of assists the home team registered while short handed.

**home\_short\_handed\_goals**

Returns an `int` of the number of goals the home team scored while short handed.

**home\_shots\_on\_goal**

Returns an `int` of the number of shots on goal the home team registered.

**home\_shutout**

Returns an `int` denoting whether or not the home team shutout the home team.

**losing\_abbr**

Returns a `string` of the losing team's abbreviation, such as 'WSH' for the Washington Capitals.

**losing\_name**

Returns a `string` of the losing team's name, such as 'Washington Capitals'.

**playoff\_round**

Returns a `string` denoting which round of the playoffs the game is a part of, such as 'Western First Round', or None if the game was played during the regular season.

**time**

Returns a `string` of the time the game started.

**winner**

Returns a `string` constant indicating whether the home or away team won.

**winning\_abbr**

Returns a `string` of the winning team's abbreviation, such as 'VEG' for the Vegas Golden Knights.

**winning\_name**

Returns a `string` of the winning team's name, such as 'Vegas Golden Knights'.

```
class sportsreference.nhl.boxscore.BoxscorePlayer(player_id, player_name,
                                                player_data)
Bases: sportsreference.nhl.player.AbstractPlayer
```

Get player stats for an individual game.

Given a player ID, such as 'zettehe01' for Henrik Zetterberg, their full name, and all associated stats from the Boxscore page in HTML format, parse the HTML and extract only the relevant stats for the specified player and assign them to readable properties.

This class inherits the `AbstractPlayer` class. As a result, all properties associated with `AbstractPlayer` can also be read directly from this class.

As this class is instantiated from within the `Boxscore` class, it should not be called directly and should instead be queried using the appropriate players properties from the `Boxscore` class.

#### Parameters

- **player\_id** (*string*) – A player's ID according to hockey-reference.com, such as 'zettehe01' for Henrik Zetterberg. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'LLLLLFFNN' where 'LLLLL' are the first 5 letters in the player's last name, 'FF', are the first 2 letters in the player's first name, and 'NN' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

#### dataframe

Returns a `pandas DataFrame` containing all other relevant properties and values for the specified game.

#### decision

Returns a `string` denoting whether the goalie won or lost the game.

#### defensive\_zone\_start\_percentage

Returns a `float` of the percentage of starts that took place in the player's defensive zone. Percentage ranges from 0-100.

#### defensive\_zone\_starts

Returns an `int` of the number of starts that took place in the player's defensive zone.

#### individual\_corsi\_for\_events

Returns an `int` of the number of individual events that impacted the player's Corsi For score during the game.

#### offensive\_zone\_starts

Returns an `int` of the number of starts that took place in the player's offensive zone.

#### on\_ice\_shot\_attempts\_against

Returns an `int` of the Corsi Against shot attempts that occurred while the player was on the ice.

#### on\_ice\_shot\_attempts\_for

Returns an `int` of the Corsi For shot attempts that occurred while the player was on the ice.

#### shifts

Returns an `int` of the number of shifts the player had on the ice during the game.

#### time\_on\_ice

Returns a `string` of the total time the player has spent on ice in the format 'MM:SS'.

**class** sportsreference.nhl.boxscore.Boxscores(*date*, *end\_date*=None)

Bases: object

Search for NHL games taking place on a particular day.

Retrieve a dictionary which contains a list of all games being played on a particular day. Output includes a link to the boxscore, and the names and abbreviations for both the home teams. If no games are played on a particular day, the list will be empty.

#### Parameters

- **date** (*datetime object*) – The date to search for any matches. The month, day, and year are required for the search, but time is not factored into the search.
- **end\_date** (*datetime object*) – Optionally specify an end date to iterate until. All boxscores starting from the date specified in the ‘date’ parameter up to and including the boxscores specified in the ‘end\_date’ parameter will be pulled. If left empty, or if ‘end\_date’ is prior to ‘date’, only the games from the day specified in the ‘date’ parameter will be saved.

#### games

Returns a dictionary object representing all of the games played on the requested day. Dictionary is in the following format:

```
{ 'date' : [ # 'date' is the string date in format 'MM-DD-YYYY'
    {
        'home_name': Name of the home team, such as 'New York
                    Rangers' (`str`),
        'home_abbr': Abbreviation for the home team, such as
                    'NYR' (`str`),
        'away_name': Name of the away team, such as 'Boston Bruins'
                    (`str`),
        'away_abbr': Abbreviation for the away team, such as 'BOS'
                    (`str`),
        'boxscore': String representing the boxscore URI, such as
                    '201702040VAN' (`str`),
        'winning_name': Full name of the winning team, such as
                    'New York Rangers' (`str`),
        'winning_abbr': Abbreviation for the winning team, such as
                    'NYR' (`str`),
        'losing_name': Full name of the losing team, such as
                    'Boston Bruins' (`str`),
        'losing_abbr': Abbreviation for the losing team, such as
                    'BOS' (`str`),
        'home_score': Integer score for the home team (`int`),
        'away_score': Integer score for the away team (`int`)
    },
    { ... },
    ...
  ]
}
```

If no games were played on ‘date’, the list for [‘date’] will be empty.

sportsreference.nhl.boxscore.nhl\_int\_property\_decorator(*func*)

## Player

The Player module contains an abstract base class that can be inherited by both the BoxscorePlayer and Player classes in the Boxscore and Roster modules, respectively. All of the properties that appear in the

`AbstractPlayer` class can be read from either of the two child classes mentioned above.

**class** `sportsreference.nhl.player.AbstractPlayer` (*player\_id*, *player\_name*, *player\_data*)  
 Bases: `object`

Get player information and stats for all seasons.

Given a player ID, such as 'zettehe01' for Henrik Zetterberg, capture all relevant stats and information like name, team, height/weight, career goals, single-season assists, penalty minutes, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on sports-reference.com.

#### Parameters

- **player\_id** (*string*) – A player's ID according to hockey-reference.com, such as 'zettehe01' for Henrik Zetterberg. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'llllffnn' where 'lllll' is the first five letters of the player's last name, 'ff' is the first two letters of the player's first name, and 'nn' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.
- **player\_name** (*string*) – A string representing the player's first and last name, such as 'Henrik Zetterberg'.
- **player\_data** (*string*) – A string representation of the player's HTML data from the Boxscore page. If the player appears in multiple tables, all of their information will appear in one single string concatenated together.

#### **assists**

Returns an `int` of the number of goals the player has assisted.

#### **blocks\_at\_even\_strength**

Returns an `int` of the number of shots the player blocks while at even strength.

#### **corsi\_for\_percentage**

Returns a `float` of the 'Corsi For' percentage, equal to `corsi_for / (corsi_for + corsi_against)`. Percentage ranges from 0-100.

#### **even\_strength\_assists**

Returns an `int` of the number of goals the player has assisted while at even strength.

#### **even\_strength\_goals**

Returns an `int` of the number of goals the player has scored at even strength.

#### **game\_winning\_goals**

Returns an `int` of the number of game-winning goals the player has scored.

#### **goals**

Returns an `int` of the number of goals the player scored.

#### **goals\_against**

Returns an `int` of the number of goals the opponent scored on the player while in goal.

#### **hits\_at\_even\_strength**

Returns an `int` of the number of hits the player makes while at even strength.

#### **name**

Returns a `string` of the player's name, such as 'Henrik Zetterberg'.

**offensive\_zone\_start\_percentage**

Returns a `float` of the percentage of faceoffs that occur in the offensive zone while the player is on ice. Percentage ranges from 0-100.

**penalties\_in\_minutes**

Returns an `int` of the number of minutes the player has served as a result of penalties.

**player\_id**

Returns a `string` of the player's ID on hockey-reference, such as 'zettehe01' for Henrik Zetterberg.

**plus\_minus**

Returns an `int` representing the relative presence the player has on the outcome of the game.

**points**

Returns an `int` of the number of points the player has gained.

**power\_play\_assists**

Returns an `int` of the number of goals the player has assisted while on a power play.

**power\_play\_goals**

Returns an `int` of the number of goals the player has scored while on a power play.

**relative\_corsi\_for\_percentage**

Returns a `float` of the player's relative 'Corsi For' percentage, equal to the difference between the player's on and off-ice Corsi For percentage.

**save\_percentage**

Returns a `float` of the percentage of shots the player has saved. Percentage ranges from 0-1.

**saves**

Returns an `int` of the number of shots the player has saved while in goal.

**shooting\_percentage**

Returns a `float` of the percentage of the player's shots that go in the goal. Percentage ranges from 0-100.

**short\_handed\_assists**

Returns an `int` of the number of goals the player has assisted while short handed.

**short\_handed\_goals**

Returns an `int` of the number of goals the player has scored while short handed.

**shots\_against**

Returns an `int` of the number of shots the opponent took while the player is in goal.

**shots\_on\_goal**

Returns an `int` of the number of shots on goal the player has made.

**shutouts**

Returns an `int` of the number of shutouts the player has registered while in goal.

## Roster

The Roster module contains detailed player information, allowing each player to be queried by their player ID using the `Player` class which has detailed information ranging from career points totals to single-season stats and player height and weight. The following is an example on collecting career information for Henrik Zetterberg:

```
from sportsreference.nhl.roster import Player

zetterberg = Player('zettehe01')
print(zetterberg.name)  # Prints 'Henrik Zetterberg'
```

(continues on next page)

(continued from previous page)

```
print(zetterberg.points)  # Prints Zetterberg's career points total
# Prints a Pandas DataFrame of all relevant Zetterberg stats per season
print(zetterberg.dataframe)
```

By default, the player's career stats are returned whenever a property is called. To get stats for a specific season, call the class instance with the season string. All future property requests will return the season-specific stats.

```
from sportsreference.nhl.roster import Player

zetterberg = Player('zettehe01')  # Currently pulling career stats
print(zetterberg.points)  # Prints Zetterberg's CAREER points total
# Prints Zetterberg's points total only for the 2017-18 season.
print(zetterberg('2017-18').points)
# Prints the number of games Zetterberg played in the 2017-18 season.
print(zetterberg.games_played)
```

After requesting single-season stats, the career stats can be requested again by calling the class without arguments or with the 'Career' string passed.

```
from sportsreference.nhl.roster import Player

zetterberg = Player('zettehe01')  # Currently pulling career stats
# Prints Zetterberg's points total only for the 2017-18 season.
print(zetterberg('2017-18').points)
print(zetterberg('Career').points)  # Prints Zetterberg's career points total
```

In addition, the Roster module also contains the Roster class which can be used to pull all players on a team's roster during a given season and creates instances of the Player class for each team member and adds them to a list to be easily queried.

```
from sportsreference.nhl.roster import Roster

detroit = Roster('DET')
for player in detroit.players:
    # Prints the name of all players who played for Houston in the most
    # recent season.
    print(player.name)
```

**class** sportsreference.nhl.roster.**Player** (*player\_id*)  
Bases: *sportsreference.nhl.player.AbstractPlayer*

Get player information and stats for all seasons.

Given a player ID, such as 'zettehe01' for Henrik Zetterberg, capture all relevant stats and information like name, team, height/weight, career goals, single-season assists, penalty minutes, and much more.

By default, the class instance will return the player's career stats, but single-season stats can be found by calling the instance with the requested season as denoted on sports-reference.com.

**Parameters** **player\_id** (*string*) – A player's ID according to hockey-reference.com, such as 'zettehe01' for Henrik Zetterberg. The player ID can be found by navigating to the player's stats page and getting the string between the final slash and the '.html' in the URL. In general, the ID is in the format 'llllffnn' where 'llll' is the first five letters of the player's last name, 'ff' is the first two letters of the player's first name, and 'nn' is a number starting at '01' for the first time that player ID has been used and increments by 1 for every successive player.

**adjusted\_assists**

Returns an int of the adjusted number of goals the player has assisted.

**adjusted\_goals**

Returns an `int` of the adjusted number of goals the player has scored.

**adjusted\_goals\_against\_average**

Returns a `float` of the adjusted goals against average for the player while in goal.

**adjusted\_goals\_created**

Returns an `int` of the adjusted number of goals the player created.

**adjusted\_points**

Returns an `int` of the adjusted number of points the player has gained.

**age**

Returns an `int` of the player's age on February 1st of the season.

**average\_time\_on\_ice**

Returns a `string` of the average time the player spends on the ice per game.

**blocks\_at\_even\_strength**

Returns an `int` of the number of shots the player blocks while at even strength.

**corsi\_against**

Returns a `float` of the player's 'Corsi Against' factor at even strength, equal to shots + blocks + misses.

**corsi\_for**

Returns a `float` of the player's 'Corsi For' factor at even strength, equal to shots + blocks + misses.

**dataframe**

Returns a `pandas DataFrame` containing all other relevant class properties and values where each index is a different season plus the career stats.

**defensive\_point\_shares**

Returns a `float` of the player's defensive point share, equal to the approximate number of points the player contributed to while on defense.

**defensive\_zone\_start\_percentage**

Returns a `float` of the percentage of faceoffs that occur in the defensive zone while the player is on ice. Percentage ranges from 0-100.

**even\_strength\_goals\_allowed**

Returns an `int` of the number of goals the player allowed in goal while at even strength.

**even\_strength\_save\_percentage**

Returns a `float` of the player's save percentage while at even strength.

**even\_strength\_shots\_faced**

Returns an `int` of the number of shots the player has faced while at even strength.

**faceoff\_losses**

Returns an `int` of the number of faceoffs the player lost.

**faceoff\_percentage**

Returns a `float` of the percentage of faceoffs the player wins. Percentage ranges from 0-100.

**faceoff\_wins**

Returns an `int` of the number of faceoffs the player won.

**fenwick\_against**

Returns an `int` of the player's 'Fenwick Against' factor at even strength, equal to shots + misses.

**fenwick\_for**

Returns an `int` of the player's 'Fenwick For' factor at even strength, equal to shots + misses.



**fenwick\_for\_percentage**

Returns a `float` of the player's 'Fenwick For' percentage, equal to `fenwick_for / (fenwick_for + fenwick_against)`. Percentage ranges from 0-100.

**games\_played**

Returns an `int` of the number of games the player participated in.

**giveaways**

Returns an `int` of the number of times the player gave the puck away to an opponent.

**goal\_against\_percentage\_relative**

Returns an `int` of the player's goals against average compared to the league average where 100 is an average player and 0 means the player saved every single shot.

**goalie\_point\_shares**

Returns a `float` of the player's point share while in goal.

**goals\_against\_average**

Returns a `float` of the average number of goals the opponent has scored per game while the player is in goal.

**goals\_against\_on\_ice**

Returns an `int` of the number of times the team has been scored on while the player is on ice.

**goals\_created**

Returns an `int` of the number of goals the player created, equal to  $(goals + assists * 0.5) * team\_goals / (team\_goals + team\_assists * 0.5)$ .

**goals\_for\_on\_ice**

Returns an `int` of the number of goals the team has scored while the player is on ice.

**goals\_saved\_above\_average**

Returns a `float` of the number of goals the player saved above the league average.

**height**

Returns a `string` of the player's height in the format "feet-inches".

**league**

Returns a `string` of the league the player's team participates in.

**losses**

Returns an `int` of the number of times the team lost while the player is in goal.

**minutes**

Returns an `int` of the total number of minutes the player has spent in goal.

**name**

Returns a `string` of the player's name, such as 'Henrik Zetterberg'.

**offensive\_point\_shares**

Returns a `float` of the player's offensive point share, equal to the approximate number of points the player contributed to while on offense.

**pdo**

Returns a `float` of the team's PDO while the player is on ice at even strength, equal to the team's shooting percentage + save percentage. Percentage ranges from 0-100.

**point\_shares**

Returns a `float` of the player's total point share, equal to the sum of the player's offensive and defensive point share.

**power\_play\_goals\_against\_on\_ice**

Returns an `int` of the total number of power play goals against while the player was on ice.

**power\_play\_goals\_allowed**

Returns an `int` of the number of goals the player allowed in goal while on a power play.

**power\_play\_goals\_for\_on\_ice**

Returns an `int` of the total number of power play goals for while the player was on ice.

**power\_play\_save\_percentage**

Returns a `float` of the player's save percentage while on a power play.

**power\_play\_shots\_faced**

Returns an `int` of the number of shots the player has faced while on a power play.

**quality\_start\_percentage**

Returns a `float` of the percentage of the player's starts that are considered quality starts while in goal. Percentage ranges from 0-1.

**quality\_starts**

Returns an `int` of the number of quality starts the player has had, equal to starting out with an in-game save percentage greater than the player's average save percentage for the year.

**really\_bad\_starts**

Returns an `int` of the number of really bad starts the player has had, equal to starting out with an in-game save percentage less than 85%.

**relative\_fenwick\_for\_percentage**

Returns a `float` of the player's relative 'Fenwick For' percentage, equal to the difference between the player's on and off-ice Fenwick For percentage.

**save\_percentage\_on\_ice**

Returns an `int` of the team's save percentage while the player is on ice.

**season**

Returns a `string` of the season in the format 'YYYY-YY', such as '2017-18'. If no season was requested, the career stats will be returned for the player and the season will default to 'Career'.

**shooting\_percentage\_on\_ice**

Returns a `float` of the team's shooting percentage while the player is on ice.

**shootout\_attempts**

Returns an `int` of the number of shootouts the player attempted.

**shootout\_goals**

Returns an `int` of the number of shootout goals the player scored.

**shootout\_misses**

Returns an `int` of the number of shootouts the player failed to score.

**shootout\_percentage**

Returns a `float` of the percentage of shootouts the player scores in. Percentage ranges from 0-100.

**short\_handed\_goals\_allowed**

Returns an `int` of the number of goals the player allowed in goal while short handed.

**short\_handed\_save\_percentage**

Returns a `float` of the player's save percentage while short handed.

**short\_handed\_shots\_faced**

Returns an `int` of the number of shots the player has faced while short handed.

**takeaways**

Returns an `int` of the number of times the player took the puck away from an opponent.

**team\_abbreviation**

Returns a `string` of the team's abbreviation, such as 'DET' for the Detroit Red Wings.

**ties\_plus\_overtime\_loss**

Returns an `int` of the number of times the team has either tied or lost in overtime or a shootout while the player is in goal.

**time\_on\_ice**

Returns an `int` of the total time the player has spent on ice in minutes.

**time\_on\_ice\_even\_strength**

Returns a `float` of the amount of time the player spent on ice in minutes while at even strength.

**total\_goals\_against\_on\_ice**

Returns an `int` of the total number of goals against while the player was on ice.

**total\_goals\_for\_on\_ice**

Returns an `int` of the total number of goals for while the player was on ice.

**total\_shots**

Returns an `int` of the total number of shots the player took regardless of them being on goal or not.

**weight**

Returns an `int` of the player's weight in pounds.

**wins**

Returns an `int` of the number of times the team won while the player is in goal.

**class** `sportsreference.nhl.roster.Roster` (*team*, *year=None*, *slim=False*)

Bases: `object`

Get stats for all players on a roster.

Request a team's roster for a given season and create instances of the `Player` class for each player, containing a detailed list of the player's statistics and information.

**Parameters**

- **team** (*string*) – The team's abbreviation, such as 'DET' for the Detroit Red Wings.
- **year** (*string (optional)*) – The 6-digit year to pull the roster from, such as '2017-18'. If left blank, defaults to the most recent season.
- **slim** (*boolean (optional)*) – Set to `True` to return a limited subset of player information including the name and player ID for each player as opposed to all of their respective stats which greatly reduces the time to return a response if just the names and IDs are desired. Defaults to `False`.

**players**

Returns a `list` of player instances for each player on the requested team's roster if the `slim` property is `False` when calling the `Roster` class. If the `slim` property is `True`, returns a `dictionary` where each key is a string of the player's ID and each value is the player's first and last name as listed on the roster page.

## Schedule

The `Schedule` module can be used to iterate over all games in a team's schedule to get game information such as the date, score, result, and more. Each game also has a link to the `Boxscore` class which has much more detailed information on the game metrics.

```
from sportsreference.nhl.schedule import Schedule

detroit_schedule = Schedule('DET')
for game in detroit_schedule:
    print(game.date)    # Prints the date the game was played
    print(game.result)  # Prints whether the team won or lost
    # Creates an instance of the Boxscore class for the game.
    boxscore = game.boxscore
```

**class** sportsreference.nhl.schedule.**Game**(*game\_data*, *year*)

Bases: object

A representation of a matchup between two teams.

Stores all relevant high-level match information for a game in a team's schedule including date, time, opponent, and result.

#### Parameters

- **game\_data** (*string*) – The row containing the specified game information.
- **year** (*string*) – The year of the current season.

#### **boxscore**

Returns an instance of the Boxscore class containing more detailed stats on the game.

#### **boxscore\_index**

Returns a *string* of the URI for a boxscore which can be used to access or index a game.

#### **corsi\_against**

Returns an *int* of the Corsi Against at Even Strength metric which equals the number of shots + blocks + misses by the opponent.

#### **corsi\_for**

Returns an *int* of the Corsi For at Even Strength metric which equals the number of shots + blocks + misses.

#### **corsi\_for\_percentage**

Returns a *float* of the percentage of control a team had of the puck which is calculated by the *corsi\_for* value divided by the sum of *corsi\_for* and *corsi\_against*. Values greater than 50.0 indicate the team had more control of the puck than their opponent. Percentage ranges from 0-100.

#### **dataframe**

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the boxscore string.

#### **dataframe\_extended**

Returns a pandas DataFrame representing the Boxscore class for the game. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property. The index for the DataFrame is the boxscore string.

#### **date**

Returns a *string* of the date the game was played, such as '2017-10-05'.

#### **datetime**

Returns a datetime object to indicate the month, day, and year the game was played at.

#### **faceoff\_losses**

Returns an *int* of the number of faceoffs the team lost at even strength.

**faceoff\_win\_percentage**

Returns a `float` of percentage of faceoffs the team won while at even strength. Percentage ranges from 0-100.

**faceoff\_wins**

Returns an `int` of the number of faceoffs the team won at even strength.

**fenwick\_against**

Returns an `int` of the Fenwick Against at Even Strength metric which equals the number of shots + misses by the opponent.

**fenwick\_for**

Returns an `int` of the Fenwick For at Even Strength metric which equals the number of shots + misses.

**fenwick\_for\_percentage**

Returns a `float` of the percentage of control a team had of the puck which is calculated by the `fenwick_for` value divided by the sum of `fenwick_for` and `fenwick_against`. Values greater than 50.0 indicate the team had more control of the puck than their opponent. Percentage ranges from 0-100.

**game**

Returns an `int` to indicate which game in the season was requested. The first game of the season returns 1.

**goals\_allowed**

Returns an `int` of the number of goals the team allowed during the game.

**goals\_scored**

Returns an `int` of the number of goals the team scored during the game.

**location**

Returns a `string` constant to indicate whether the game was played at home or away.

**offensive\_zone\_start\_percentage**

Returns a `float` of the percentage of stats that took place in the offensive half. Value is calculated by the number of offensive zone starts divided by the sum of offensive zone starts and defensive zone starts. Percentage ranges from 0-100.

**opp\_penalties\_in\_minutes**

Returns an `int` of the total number of minutes the opponent served for penalties.

**opp\_power\_play\_goals**

Returns an `int` of the number of power play goals the opponent scored.

**opp\_power\_play\_opportunities**

Returns an `int` of the number of power play opportunities the opponent had.

**opp\_short\_handed\_goals**

Returns an `int` of the number of shorthanded goals the opponent scored.

**opp\_shots\_on\_goal**

Returns an `int` of the total number of shots on goal the opponent registered.

**opponent\_abbr**

Returns a `string` of the opponent's 3-letter abbreviation, such as 'NYR' for the New York Rangers.

**opponent\_name**

Returns a `string` of the opponent's name, such as 'New York Rangers'.

**overtime**

Returns an `int` of the number of overtimes that were played during the game, or an `int` constant if the game went to a shootout.

**pdo**

Returns a `float` of the team's PDO at Even Strength metric which is calculated by the sum of the shooting percentage and save percentage. Percentage ranges from 0-100.

**penalties\_in\_minutes**

Returns an `int` of the total number of minutes the team served for penalties.

**power\_play\_goals**

Returns an `int` of the number of power play goals the team scored.

**power\_play\_opportunities**

Returns an `int` of the number of power play opportunities the team had.

**result**

Returns a `string` constant to indicate whether the team lost in regulation, lost in overtime, or won.

**short\_handed\_goals**

Returns an `int` of the number of shorthanded goals the team scored.

**shots\_on\_goal**

Returns an `int` of the total number of shots on goal the team registered.

**class** `sportsreference.nhl.schedule.Schedule` (*abbreviation*, *year=None*)

Bases: `object`

An object of the given team's schedule.

Generates a team's schedule for the season including wins, losses, and scores if applicable.

**Parameters**

- **abbreviation** (*string*) – A team's short name, such as 'NYR' for the New York Rangers.
- **year** (*string (optional)*) – The requested year to pull stats from.

**dataframe**

Returns a pandas DataFrame where each row is a representation of the Game class. Rows are indexed by the boxscore string.

**dataframe\_extended**

Returns a pandas DataFrame where each row is a representation of the Boxscore class for every game in the schedule. Rows are indexed by the boxscore string. This property provides much richer context for the selected game, but takes longer to process compared to the lighter 'dataframe' property.

## Teams

The Teams module exposes information for all NHL teams including the team name and abbreviation, the number of games they won during the season, the total number of shots on goal, and much more.

```
from sportsreference.nhl.teams import Teams

teams = Teams()
for team in teams:
    print(team.name)    # Prints the team's name
    print(team.shots_on_goal)  # Prints the team's total shots on goal
```

Each Team instance contains a link to the `Schedule` class which enables easy iteration over all games for a particular team. A Pandas DataFrame can also be queried to easily grab all stats for all games.

```

from sportsreference.nhl.teams import Teams

teams = Teams()
for team in teams:
    schedule = team.schedule # Returns a Schedule instance for each team
    # Returns a Pandas DataFrame of all metrics for all game Boxscores for
    # a season.
    df = team.schedule.dataframe_extended

```

Lastly, each Team instance also contains a link to the Roster class which enables players from the team to be easily queried. Each Roster instance contains detailed stats and information for each player on the team.

```

from sportsreference.nhl.teams import Teams

teams = Teams()
for team in teams:
    # Creates an instance of the roster class for each player on the team.
    roster = team.roster
    for player in roster.players:
        print(player.name) # Prints the name of each player on the team.

```

```

class sportsreference.nhl.teams.Team(team_data, rank, year=None)
    Bases: object

```

An object containing all of a team's season information.

Finds and parses all team stat information and identifiers, such as rank, name, and abbreviation, and sets them as properties which can be directly read from for easy reference.

#### Parameters

- **team\_data** (*string*) – A string containing all of the rows of stats for a given team. If multiple tables are being referenced, this will be comprised of multiple rows in a single string.
- **rank** (*int*) – A team's position in the league based on the number of points they obtained during the season.
- **year** (*string (optional)*) – The requested year to pull stats from.

#### abbreviation

Returns a *string* of the team's abbreviation, such as 'DET' for the Detroit Red Wings.

#### average\_age

Returns a *float* of the average age of all players on the team, weighted by their time on ice.

#### dataframe

Returns a pandas DataFrame containing all other class properties and values. The index for the DataFrame is the string abbreviation of the team, such as 'DET'.

#### games\_played

Returns an *int* of the total number of games the team has played in the season.

#### goals\_against

Returns an *int* of the total number of goals opponents scored against the team during the season.

#### goals\_for

Returns an *int* of the total number of goals a team scored during the season.

#### losses

Returns an *int* of the total number of losses the team had in the season.

**name**

Returns a `string` of the team's full name, such as 'Detroit Red Wings'.

**overtime\_losses**

Returns an `int` of the total number of overtime losses the team had in the season.

**pdo\_at\_even\_strength**

Returns a `float` of the PDO at even strength which equates to the shooting percentage plus the save percentage.

**penalty\_killing\_percentage**

Returns a `float` denoting the percentage of power plays that have been successfully defended without a goal being conceded. Percentage ranges from 0-100.

**points**

Returns an `int` of the total number of points the team gained in the season.

**points\_percentage**

Returns a `float` denoting the percentage of points gained divided by the maximum possible points available during the season. Percentage ranges from 0-1.

**power\_play\_goals**

Returns an `int` of the total number of power play goals scored.

**power\_play\_goals\_against**

Returns an `int` of the total number of power play goals conceded.

**power\_play\_opportunities**

Returns an `int` of the total number of power play opportunities for a team during the season.

**power\_play\_opportunities\_against**

Returns an `int` of the total number of power play opportunities for the opponents during the season.

**power\_play\_percentage**

Returns a `float` denoting the percentage of power play opportunities where the team has scored. Percentage ranges from 0-100.

**rank**

Returns an `int` of the team's rank based on the number of points they obtained in the season.

**roster**

Returns an instance of the Roster class containing all players for the team during the season with all career stats.

**save\_percentage**

Returns a `float` denoting the percentage of shots the team has saved during the season. Percentage ranges from 0-1.

**schedule**

Returns an instance of the Schedule class containing the team's complete schedule for the season.

**shooting\_percentage**

Returns a `float` denoting the percentage of shots to goals during the season. Percentage ranges from 0-100.

**short\_handed\_goals**

Returns an `int` of the number of short handed goals the team has scored during the season.

**short\_handed\_goals\_against**

Returns an `int` of the number of short handed goals the team has conceded during the season.

**shots\_against**

Returns an `int` of the total number of shots on goal the team's opponents made during the season.



**shots\_on\_goal**

Returns an `int` of the total number of shots on goal the team made during the season.

**simple\_rating\_system**

Returns a `float` which takes into account the average goal differential vs a team's strength of schedule. The league average evaluates to 0.0. Teams which have a positive score are comparatively stronger than average while teams with a negative score are weaker.

**strength\_of\_schedule**

Returns a `float` denoting a team's strength of schedule, based on goals scores and conceded. Higher values result in more challenging schedules while 0.0 is an average schedule.

**total\_goals\_per\_game**

Returns a `float` for the average number of goals scored per game.

**wins**

Returns an `int` of the total number of wins the team had in the season.

**class** `sportsreference.nhl.teams.Teams` (*year=None*)

Bases: `object`

A list of all NHL teams and their stats in a given year.

Finds and retrieves a list of all NHL teams from [www.hockey-reference.com](http://www.hockey-reference.com) and creates a `Team` instance for every team that participated in the league in a given year. The `Team` class comprises a list of all major stats and a few identifiers for the requested season.

**Parameters** *year* (*string (optional)*) – The requested year to pull stats from.

**dataframes**

Returns a pandas `DataFrame` where each row is a representation of the `Team` class. Rows are indexed by the team abbreviation.

## 1.6.2 Examples

Thanks to the broad range of metrics that are pulled from [sports-reference.com](http://sports-reference.com), there are multiple ways you can use the *sportsreference* package. This page has multiple examples beyond those listed on the home page to demonstrate some cool things you can do which leverage the tool. This page is by no means exhaustive and the examples aren't necessarily the most efficient in the hope of providing the most clarity.

In general, most examples shown for a specific sport are applicable for all sports currently supported by *sportsreference*.

### 1.6.2.1 Finding Tallest Players

For each team, find the tallest player on the roster and print out their name and height in inches.

```
from sportsreference.nba.teams import Teams

def get_height_in_inches(height):
    feet, inches = height.split('-')
    return int(feet) * 12 + int(inches)

def print_tallest_player(team_heights):
    tallest_player = max(team_heights, key=team_heights.get)
    print('%s: %s in.' % (tallest_player, team_heights[tallest_player]))

for team in Teams():
```

(continues on next page)

(continued from previous page)

```
print('=' * 80)
print(team.name)
team_heights = {}
for player in team.roster.players:
    height = get_height_in_inches(player.height)
    team_heights[player.name] = height
print_tallest_player(team_heights)
```

### 1.6.2.2 Writing To CSV and Pickle

To prevent re-pulling data from datasets that won't change, such as completed games with fixed statistics, the pandas DataFrame can be saved to the local filesystem for re-use later on. Two common file types for this are CSV files and the high-performing Pickle files. CSV files are a common file type that many tools and editors support and save an interpretation of the DataFrame, while a Pickle file is a special file that saves the DataFrame exactly as-is. Pickle files are faster to read and write compared to CSV files and don't pose a risk of missing or altered data compared to CSV files.

Save the combined stats for each team to both a CSV and Pickle file.

```
from sportsreference.ncaab.teams import Teams

for team in Teams():
    team.dataframe.to_csv('%s.csv' % team.abbreviation.lower())
    team.dataframe.to_pickle('%s.pkl' % team.abbreviation.lower())
```

### 1.6.2.3 Finding Top Win Percentage By Year

For each year in a range, find the team with the most wins during the season and print their name and the win total.

```
from sportsreference.mlb.teams import Teams

def print_most_wins(year, wins):
    most_wins = max(wins, key=wins.get)
    print('%s: %s - %s' % (year, wins[most_wins], most_wins))

for year in range(2000, 2019):
    wins = {}
    for team in Teams(year):
        wins[team.name] = team.wins
    print_most_wins(year, wins)
```

## 1.6.3 Installation

The easiest way to install *sportsreference* is by downloading the latest released binary from PyPI using PIP. For instructions on installing PIP, visit [PyPA.io](https://pyPA.io) for detailed steps on installing the package manager for your local environment.

Next, run:

```
pip install sportsreference
```

to download and install the latest official release of *sportsreference* on your machine. You now have the latest stable version of *sportsreference* installed and can begin using it following the examples!

If the bleeding-edge version of *sportsreference* is desired, clone this repository using git and install all of the package requirements with PIP:

```
git clone https://github.com/roclark/sportsreference
cd sportsreference
pip install -r requirements.txt
```

Once complete, create a Python wheel for your default version of Python by running the following command:

```
python setup.py sdist bdist_wheel
```

This will create a *.whl* file in the *dist* directory which can be installed with the following command:

```
pip install dist/*.whl
```

### 1.6.4 Testing

Sportsreference contains a testing suite which aims to test all major portions of code for proper functionality. To run the test suite against your environment, ensure all of the requirements are installed by running:

```
pip install -r requirements.txt
```

Next, start the tests by running `py.test` while optionally including coverage flags which identify the amount of production code covered by the testing framework:

```
py.test --cov=sportsreference --cov-report term-missing tests/
```

If the tests were successful, it will return a green line will show a message at the end of the output similar to the following:

```
===== 380 passed in 245.56 seconds =====
```

If a test failed, it will show the number of failed and what went wrong within the test output. If that's the case, ensure you have the latest version of code and are in a supported environment. Otherwise, create an issue on GitHub to attempt to get the issue resolved.



## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### S

- `sportsreference.mlb.boxscore`, 5
- `sportsreference.mlb.player`, 12
- `sportsreference.mlb.roster`, 14
- `sportsreference.mlb.schedule`, 20
- `sportsreference.mlb.teams`, 22
- `sportsreference.nba.boxscore`, 29
- `sportsreference.nba.player`, 36
- `sportsreference.nba.roster`, 39
- `sportsreference.nba.schedule`, 43
- `sportsreference.nba.teams`, 46
- `sportsreference.ncaab.boxscore`, 49
- `sportsreference.ncaab.conferences`, 57
- `sportsreference.ncaab.player`, 58
- `sportsreference.ncaab.rankings`, 60
- `sportsreference.ncaab.roster`, 62
- `sportsreference.ncaab.schedule`, 64
- `sportsreference.ncaab.teams`, 67
- `sportsreference.ncaaf.boxscore`, 73
- `sportsreference.ncaaf.conferences`, 78
- `sportsreference.ncaaf.player`, 79
- `sportsreference.ncaaf.rankings`, 82
- `sportsreference.ncaaf.roster`, 84
- `sportsreference.ncaaf.schedule`, 87
- `sportsreference.ncaaf.teams`, 90
- `sportsreference.nfl.boxscore`, 94
- `sportsreference.nfl.player`, 99
- `sportsreference.nfl.roster`, 103
- `sportsreference.nfl.schedule`, 109
- `sportsreference.nfl.teams`, 113
- `sportsreference.nhl.boxscore`, 116
- `sportsreference.nhl.player`, 121
- `sportsreference.nhl.roster`, 123
- `sportsreference.nhl.schedule`, 128
- `sportsreference.nhl.teams`, 131





## A

- abbreviation (*sportsreference.mlb.teams.Team attribute*), 23
- abbreviation (*sportsreference.nba.teams.Team attribute*), 46
- abbreviation (*sportsreference.ncaab.teams.Team attribute*), 67
- abbreviation (*sportsreference.ncaaf.teams.Team attribute*), 90
- abbreviation (*sportsreference.nfl.teams.Team attribute*), 113
- abbreviation (*sportsreference.nhl.teams.Team attribute*), 131
- AbstractPlayer (class in *sportsreference.mlb.player*), 12
- AbstractPlayer (class in *sportsreference.nba.player*), 36
- AbstractPlayer (class in *sportsreference.ncaab.player*), 58
- AbstractPlayer (class in *sportsreference.ncaaf.player*), 79
- AbstractPlayer (class in *sportsreference.nfl.player*), 99
- AbstractPlayer (class in *sportsreference.nhl.player*), 121
- adjusted\_assists (*sportsreference.nhl.roster.Player attribute*), 123
- adjusted\_goals (*sportsreference.nhl.roster.Player attribute*), 123
- adjusted\_goals\_against\_average (*sportsreference.nhl.roster.Player attribute*), 124
- adjusted\_goals\_created (*sportsreference.nhl.roster.Player attribute*), 124
- adjusted\_net\_yards\_per\_attempt\_index (*sportsreference.nfl.roster.Player attribute*), 103
- adjusted\_net\_yards\_per\_pass\_attempt (*sportsreference.nfl.roster.Player attribute*), 103
- adjusted\_points (*sportsreference.nhl.roster.Player attribute*), 124
- adjusted\_yards\_per\_attempt (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 79
- adjusted\_yards\_per\_attempt (*sportsreference.ncaaf.roster.Player attribute*), 84
- adjusted\_yards\_per\_attempt (*sportsreference.nfl.roster.Player attribute*), 103
- adjusted\_yards\_per\_attempt\_index (*sportsreference.nfl.roster.Player attribute*), 103
- age (*sportsreference.nhl.roster.Player attribute*), 124
- all\_purpose\_yards (*sportsreference.nfl.roster.Player attribute*), 103
- and\_ones (*sportsreference.nba.roster.Player attribute*), 39
- approximate\_value (*sportsreference.nfl.roster.Player attribute*), 103
- arena (*sportsreference.ncaab.schedule.Game attribute*), 65
- arena (*sportsreference.nhl.boxscore.Boxscore attribute*), 116
- assist\_percentage (*sportsreference.nba.player.AbstractPlayer attribute*), 36
- assist\_percentage (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58
- assist\_percentage (*sportsreference.ncaab.teams.Team attribute*), 67
- assists (*sportsreference.mlb.player.AbstractPlayer attribute*), 12
- assists (*sportsreference.mlb.roster.Player attribute*), 15
- assists (*sportsreference.nba.player.AbstractPlayer attribute*), 36
- assists (*sportsreference.nba.teams.Team attribute*), 46
- assists (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58
- assists (*sportsreference.ncaab.teams.Team attribute*),

67  
assists (*sportsreference.nhl.player.AbstractPlayer attribute*), 121  
assists\_on\_tackles (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 79  
assists\_on\_tackles (*sportsreference.ncaaf.roster.Player attribute*), 84  
assists\_on\_tackles (*sportsreference.nfl.player.AbstractPlayer attribute*), 100  
assists\_on\_tackles (*sportsreference.nfl.roster.Player attribute*), 103  
at\_bats (*sportsreference.mlb.player.AbstractPlayer attribute*), 12  
at\_bats (*sportsreference.mlb.roster.Player attribute*), 15  
at\_bats (*sportsreference.mlb.teams.Team attribute*), 23  
attempted\_passes (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 80  
attempted\_passes (*sportsreference.ncaaf.roster.Player attribute*), 84  
attempted\_passes (*sportsreference.nfl.player.AbstractPlayer attribute*), 100  
attempted\_passes (*sportsreference.nfl.roster.Player attribute*), 104  
attendance (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
attendance (*sportsreference.mlb.schedule.Game attribute*), 20  
attendance (*sportsreference.nfl.boxscore.Boxscore attribute*), 94  
attendance (*sportsreference.nhl.boxscore.Boxscore attribute*), 116  
average\_age (*sportsreference.nhl.teams.Team attribute*), 131  
average\_batter\_age (*sportsreference.mlb.teams.Team attribute*), 23  
average\_kickoff\_return\_yards (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 75  
average\_kickoff\_return\_yards (*sportsreference.nfl.boxscore.BoxscorePlayer attribute*), 98  
average\_leverage\_index (*sportsreference.mlb.boxscore.BoxscorePlayer attribute*), 10  
average\_leverage\_index\_pitcher (*sportsreference.mlb.boxscore.BoxscorePlayer attribute*), 10  
average\_pitcher\_age (*sportsreference.mlb.teams.Team attribute*), 23  
average\_punt\_return\_yards (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 75  
average\_time\_on\_ice (*sportsreference.nhl.roster.Player attribute*), 124  
away\_abbreviation (*sportsreference.nfl.boxscore.Boxscore attribute*), 94  
away\_assist\_percentage (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_assist\_percentage (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_assists (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_assists (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_assists (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_assists (*sportsreference.nhl.boxscore.Boxscore attribute*), 116  
away\_at\_bats (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_average\_leverage\_index (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_base\_out\_runs\_added (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_base\_out\_runs\_saved (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_bases\_on\_balls (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_batting\_average (*sportsreference.mlb.boxscore.Boxscore attribute*), 5  
away\_block\_percentage (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_block\_percentage (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_blocks (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_blocks (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_defensive\_rating (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_defensive\_rating (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_defensive\_rebound\_percentage (*sportsreference.nba.boxscore.Boxscore attribute*), 29  
away\_defensive\_rebound\_percentage (*sportsreference.ncaab.boxscore.Boxscore attribute*), 50  
away\_defensive\_rebounds (*sportsreference*

<i>ence.nba.boxscore.Boxscore attribute</i> ), 30	away_free_throw_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50
away_defensive_rebounds ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_free_throws ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30
away_earned_runs ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 5	away_free_throws ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50
away_effective_field_goal_percentage ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_fumbles ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 73
away_effective_field_goal_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_fumbles ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94
away_even_strength_assists ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 116	away_fumbles_lost ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 73
away_even_strength_goals ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 116	away_fumbles_lost ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94
away_field_goal_attempts ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_game_score ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 5
away_field_goal_attempts ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_game_winning_goals ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 116
away_field_goal_percentage ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_goals ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 117
away_field_goal_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_grounded_balls ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_field_goals ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_hits ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_field_goals ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_home_runs ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_first_downs ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 73	away_inherited_runners ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_first_downs ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94	away_inherited_score ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_fly_balls ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 5	away_innings_pitched ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_fourth_down_attempts ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94	away_interceptions ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 73
away_fourth_down_conversions ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94	away_interceptions ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 94
away_free_throw_attempt_rate ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_line_drives ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 6
away_free_throw_attempt_rate ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_losses ( <i>sportsreference.mlb.teams.Team attribute</i> ), 23
away_free_throw_attempts ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_losses ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30
away_free_throw_attempts ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50	away_losses ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 50
away_free_throw_percentage ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30	away_losses ( <i>sportsreference.ncaab.teams.Team attribute</i> ), 67
	away_minutes_played ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 30

away_minutes_played	(sportsreference.ncaab.boxscore.Boxscore attribute), 50	away_minutes_played	(sportsreference.ncaab.boxscore.Boxscore attribute), 51
away_net_pass_yards	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_pitches	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_offensive_rating	(sportsreference.nba.boxscore.Boxscore attribute), 30	away_plate_appearances	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_offensive_rating	(sportsreference.ncaab.boxscore.Boxscore attribute), 51	away_players	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_offensive_rebound_percentage	(sportsreference.nba.boxscore.Boxscore attribute), 30	away_players	(sportsreference.nba.boxscore.Boxscore attribute), 30
away_offensive_rebound_percentage	(sportsreference.ncaab.boxscore.Boxscore attribute), 51	away_players	(sportsreference.ncaab.boxscore.Boxscore attribute), 51
away_offensive_rebounds	(sportsreference.nba.boxscore.Boxscore attribute), 30	away_players	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73
away_offensive_rebounds	(sportsreference.ncaab.boxscore.Boxscore attribute), 51	away_players	(sportsreference.nfl.boxscore.Boxscore attribute), 95
away_on_base_percentage	(sportsreference.mlb.boxscore.Boxscore attribute), 6	away_players	(sportsreference.nhl.boxscore.Boxscore attribute), 117
away_on_base_plus	(sportsreference.mlb.boxscore.Boxscore attribute), 6	away_points	(sportsreference.nba.boxscore.Boxscore attribute), 30
away_pass_attempts	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73	away_points	(sportsreference.ncaab.boxscore.Boxscore attribute), 51
away_pass_attempts	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_points	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73
away_pass_completions	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73	away_points	(sportsreference.nfl.boxscore.Boxscore attribute), 95
away_pass_completions	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_points	(sportsreference.nhl.boxscore.Boxscore attribute), 117
away_pass_touchdowns	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73	away_power_play_assists	(sportsreference.nhl.boxscore.Boxscore attribute), 117
away_pass_touchdowns	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_power_play_goals	(sportsreference.nhl.boxscore.Boxscore attribute), 117
away_pass_yards	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73	away_putouts	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_pass_yards	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_ranking	(sportsreference.ncaab.boxscore.Boxscore attribute), 51
away_penalties	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73	away_rbi	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_penalties	(sportsreference.nfl.boxscore.Boxscore attribute), 95	away_record	(sportsreference.mlb.teams.Team attribute), 23
away_penalties_in_minutes	(sportsreference.nhl.boxscore.Boxscore attribute), 117	away_runs	(sportsreference.mlb.boxscore.Boxscore attribute), 6
away_personal_fouls	(sportsreference.nba.boxscore.Boxscore attribute), 30	away_rush_attempts	(sportsreference.ncaaf.boxscore.Boxscore attribute), 73
away_personal_fouls	(sportsreference	away_rush_attempts	(sportsreference.nfl.boxscore.Boxscore attribute), 95
		away_rush_touchdowns	(sportsreference.ncaaf.boxscore.Boxscore attribute),

- 73
- away\_rush\_touchdowns (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_rush\_yards (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 73
- away\_rush\_yards (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_save\_percentage (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_saves (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_shooting\_percentage (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_short\_handed\_assists (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_short\_handed\_goals (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_shots\_on\_goal (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_shutout (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- away\_slugging\_percentage (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_steal\_percentage (*sportsreference.nba.boxscore.Boxscore* attribute), 30
- away\_steal\_percentage (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_steals (*sportsreference.nba.boxscore.Boxscore* attribute), 30
- away\_steals (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_strikeouts (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_strikes (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_strikes\_by\_contact (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_strikes\_looking (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_strikes\_swinging (*sportsreference.mlb.boxscore.Boxscore* attribute), 6
- away\_third\_down\_attempts (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_third\_down\_conversions (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_three\_point\_attempt\_rate (*sportsreference.nba.boxscore.Boxscore* attribute), 30
- away\_three\_point\_attempt\_rate (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_three\_point\_field\_goal\_attempts (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_three\_point\_field\_goal\_attempts (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_three\_point\_field\_goal\_percentage (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_three\_point\_field\_goal\_percentage (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_three\_point\_field\_goals (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_three\_point\_field\_goals (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_time\_of\_possession (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_times\_sacked (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_total\_rebound\_percentage (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_total\_rebound\_percentage (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_total\_rebounds (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_total\_rebounds (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_total\_yards (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 74
- away\_total\_yards (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_true\_shooting\_percentage (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_true\_shooting\_percentage (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_turnover\_percentage (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_turnover\_percentage (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_turnovers (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- away\_turnovers (*sportsreference.ncaab.boxscore.Boxscore* attribute), 51
- away\_turnovers (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 74
- away\_turnovers (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- away\_two\_point\_field\_goal\_attempts (*sportsreference.nba.boxscore.Boxscore* attribute), 31



- tribute), 31
- away\_two\_point\_field\_goal\_attempts (sportsreference.ncaab.boxscore.Boxscore attribute), 51
- away\_two\_point\_field\_goal\_percentage (sportsreference.nba.boxscore.Boxscore attribute), 31
- away\_two\_point\_field\_goal\_percentage (sportsreference.ncaab.boxscore.Boxscore attribute), 52
- away\_two\_point\_field\_goals (sportsreference.nba.boxscore.Boxscore attribute), 31
- away\_two\_point\_field\_goals (sportsreference.ncaab.boxscore.Boxscore attribute), 52
- away\_unknown\_bat\_type (sportsreference.mlb.boxscore.Boxscore attribute), 7
- away\_win\_percentage (sportsreference.ncaab.boxscore.Boxscore attribute), 52
- away\_win\_probability\_added (sportsreference.mlb.boxscore.Boxscore attribute), 7
- away\_win\_probability\_by\_pitcher (sportsreference.mlb.boxscore.Boxscore attribute), 7
- away\_win\_probability\_for\_offensive\_players (sportsreference.mlb.boxscore.Boxscore attribute), 7
- away\_win\_probability\_subtracted (sportsreference.mlb.boxscore.Boxscore attribute), 7
- away\_wins (sportsreference.mlb.teams.Team attribute), 23
- away\_wins (sportsreference.nba.boxscore.Boxscore attribute), 31
- away\_wins (sportsreference.ncaab.boxscore.Boxscore attribute), 52
- away\_wins (sportsreference.ncaab.teams.Team attribute), 67
- away\_yards\_from\_penalties (sportsreference.ncaaf.boxscore.Boxscore attribute), 74
- away\_yards\_from\_penalties (sportsreference.nfl.boxscore.Boxscore attribute), 95
- away\_yards\_lost\_from\_sacks (sportsreference.nfl.boxscore.Boxscore attribute), 95
- B**
- balks (sportsreference.mlb.roster.Player attribute), 15
- balks (sportsreference.mlb.teams.Team attribute), 23
- base\_out\_runs\_added (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 10
- base\_out\_runs\_saved (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 10
- bases\_on\_balls (sportsreference.mlb.player.AbstractPlayer attribute), 13
- bases\_on\_balls (sportsreference.mlb.roster.Player attribute), 15
- bases\_on\_balls (sportsreference.mlb.teams.Team attribute), 23
- bases\_on\_balls\_given (sportsreference.mlb.player.AbstractPlayer attribute), 13
- bases\_on\_balls\_given (sportsreference.mlb.roster.Player attribute), 15
- bases\_on\_balls\_given\_per\_nine\_innings (sportsreference.mlb.roster.Player attribute), 15
- bases\_on\_walks\_given (sportsreference.mlb.teams.Team attribute), 23
- bases\_on\_walks\_given\_per\_nine\_innings (sportsreference.mlb.teams.Team attribute), 23
- batters\_faced (sportsreference.mlb.player.AbstractPlayer attribute), 13
- batters\_faced (sportsreference.mlb.teams.Team attribute), 23
- batters\_struckout\_per\_nine\_innings (sportsreference.mlb.roster.Player attribute), 15
- batting\_average (sportsreference.mlb.player.AbstractPlayer attribute), 13
- batting\_average (sportsreference.mlb.roster.Player attribute), 15
- batting\_average (sportsreference.mlb.teams.Team attribute), 23
- birth\_date (sportsreference.mlb.roster.Player attribute), 15
- birth\_date (sportsreference.nba.roster.Player attribute), 39
- birth\_date (sportsreference.nfl.roster.Player attribute), 104
- block\_percentage (sportsreference.nba.player.AbstractPlayer attribute), 36
- block\_percentage (sportsreference.ncaab.player.AbstractPlayer attribute), 58
- block\_percentage (sportsreference.ncaab.teams.Team attribute), 67
- blocked\_punts (sportsreference.nfl.roster.Player attribute), 104
- blocking\_fouls (sportsreference.nba.roster.Player attribute), 39
- blocks (sportsreference.nba.player.AbstractPlayer attribute), 36

- blocks (*sportsreference.nba.teams.Team* attribute), 46
- blocks (*sportsreference.ncaab.player.AbstractPlayer* attribute), 58
- blocks (*sportsreference.ncaab.teams.Team* attribute), 67
- blocks\_at\_even\_strength (*sportsreference.nhl.player.AbstractPlayer* attribute), 121
- blocks\_at\_even\_strength (*sportsreference.nhl.roster.Player* attribute), 124
- box\_plus\_minus (*sportsreference.nba.player.AbstractPlayer* attribute), 36
- box\_plus\_minus (*sportsreference.ncaab.roster.Player* attribute), 63
- Boxscore (*class in sportsreference.mlb.boxscore*), 5
- Boxscore (*class in sportsreference.nba.boxscore*), 29
- Boxscore (*class in sportsreference.ncaab.boxscore*), 49
- Boxscore (*class in sportsreference.ncaaf.boxscore*), 73
- Boxscore (*class in sportsreference.nfl.boxscore*), 94
- Boxscore (*class in sportsreference.nhl.boxscore*), 116
- boxscore (*sportsreference.mlb.schedule.Game* attribute), 20
- boxscore (*sportsreference.nba.schedule.Game* attribute), 44
- boxscore (*sportsreference.ncaab.schedule.Game* attribute), 65
- boxscore (*sportsreference.ncaaf.schedule.Game* attribute), 88
- boxscore (*sportsreference.nfl.schedule.Game* attribute), 110
- boxscore (*sportsreference.nhl.schedule.Game* attribute), 128
- boxscore\_index (*sportsreference.mlb.schedule.Game* attribute), 20
- boxscore\_index (*sportsreference.nba.schedule.Game* attribute), 44
- boxscore\_index (*sportsreference.ncaab.schedule.Game* attribute), 65
- boxscore\_index (*sportsreference.ncaaf.schedule.Game* attribute), 88
- boxscore\_index (*sportsreference.nfl.schedule.Game* attribute), 110
- boxscore\_index (*sportsreference.nhl.schedule.Game* attribute), 128
- BoxscorePlayer (*class in sportsreference.mlb.boxscore*), 9
- BoxscorePlayer (*class in sportsreference.nba.boxscore*), 34
- BoxscorePlayer (*class in sportsreference.ncaab.boxscore*), 54
- BoxscorePlayer (*class in sportsreference.ncaaf.boxscore*), 75
- BoxscorePlayer (*class in sportsreference.nfl.boxscore*), 97
- BoxscorePlayer (*class in sportsreference.nhl.boxscore*), 118
- Boxscores (*class in sportsreference.mlb.boxscore*), 11
- Boxscores (*class in sportsreference.nba.boxscore*), 35
- Boxscores (*class in sportsreference.ncaab.boxscore*), 55
- Boxscores (*class in sportsreference.ncaaf.boxscore*), 76
- Boxscores (*class in sportsreference.nfl.boxscore*), 98
- Boxscores (*class in sportsreference.nhl.boxscore*), 119
- ## C
- catch\_percentage (*sportsreference.nfl.roster.Player* attribute), 104
- center\_percentage (*sportsreference.nba.roster.Player* attribute), 39
- combined\_tackles (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute), 98
- complete (*sportsreference.ncaab.rankings.Rankings* attribute), 61
- complete (*sportsreference.ncaaf.rankings.Rankings* attribute), 82
- complete\_game\_shutouts (*sportsreference.mlb.teams.Team* attribute), 23
- complete\_games (*sportsreference.mlb.roster.Player* attribute), 15
- complete\_games (*sportsreference.mlb.teams.Team* attribute), 23
- completed\_passes (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80
- completed\_passes (*sportsreference.ncaaf.roster.Player* attribute), 84
- completed\_passes (*sportsreference.nfl.player.AbstractPlayer* attribute), 100
- completed\_passes (*sportsreference.nfl.roster.Player* attribute), 104
- completion\_percentage\_index (*sportsreference.nfl.roster.Player* attribute), 104
- Conference (*class in sportsreference.ncaab.conferences*), 57
- Conference (*class in sportsreference.ncaaf.conferences*), 78
- conference (*sportsreference.ncaab.roster.Player* attribute), 63
- conference (*sportsreference.ncaab.teams.Team* attribute), 67
- conference (*sportsreference.ncaaf.teams.Team* attribute), 90
- conference\_losses (*sportsreference.ncaab.teams.Team* attribute), 67

conference\_losses (*sportsreference.ncaaf.teams.Team* attribute), 90

conference\_win\_percentage (*sportsreference.ncaaf.teams.Team* attribute), 90

conference\_wins (*sportsreference.ncaab.teams.Team* attribute), 67

conference\_wins (*sportsreference.ncaaf.teams.Team* attribute), 90

Conferences (class in *sportsreference.ncaab.conferences*), 57

Conferences (class in *sportsreference.ncaaf.conferences*), 78

conferences (*sportsreference.ncaab.conferences.Conferences* attribute), 57

conferences (*sportsreference.ncaaf.conferences.Conferences* attribute), 79

contract (*sportsreference.mlb.roster.Player* attribute), 15

contract (*sportsreference.nba.roster.Player* attribute), 39

corsi\_against (*sportsreference.nhl.roster.Player* attribute), 124

corsi\_against (*sportsreference.nhl.schedule.Game* attribute), 128

corsi\_for (*sportsreference.nhl.roster.Player* attribute), 124

corsi\_for (*sportsreference.nhl.schedule.Game* attribute), 128

corsi\_for\_percentage (*sportsreference.nhl.player.AbstractPlayer* attribute), 121

corsi\_for\_percentage (*sportsreference.nhl.schedule.Game* attribute), 128

current (*sportsreference.ncaab.rankings.Rankings* attribute), 61

current (*sportsreference.ncaaf.rankings.Rankings* attribute), 83

current\_extended (*sportsreference.ncaab.rankings.Rankings* attribute), 61

current\_extended (*sportsreference.ncaaf.rankings.Rankings* attribute), 83

**D**

dataframe (*sportsreference.mlb.boxscore.Boxscore* attribute), 7

dataframe (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 10

dataframe (*sportsreference.mlb.roster.Player* attribute), 15

dataframe (*sportsreference.mlb.schedule.Game* attribute), 20

dataframe (*sportsreference.mlb.schedule.Schedule* attribute), 22

dataframe (*sportsreference.mlb.teams.Team* attribute), 23

dataframe (*sportsreference.nba.boxscore.Boxscore* attribute), 31

dataframe (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 34

dataframe (*sportsreference.nba.roster.Player* attribute), 39

dataframe (*sportsreference.nba.schedule.Game* attribute), 44

dataframe (*sportsreference.nba.schedule.Schedule* attribute), 45

dataframe (*sportsreference.nba.teams.Team* attribute), 46

dataframe (*sportsreference.ncaab.boxscore.Boxscore* attribute), 52

dataframe (*sportsreference.ncaab.boxscore.BoxscorePlayer* attribute), 55

dataframe (*sportsreference.ncaab.roster.Player* attribute), 63

dataframe (*sportsreference.ncaab.schedule.Game* attribute), 65

dataframe (*sportsreference.ncaab.schedule.Schedule* attribute), 66

dataframe (*sportsreference.ncaab.teams.Team* attribute), 68

dataframe (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 74

dataframe (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76

dataframe (*sportsreference.ncaaf.roster.Player* attribute), 84

dataframe (*sportsreference.ncaaf.schedule.Game* attribute), 88

dataframe (*sportsreference.ncaaf.schedule.Schedule* attribute), 89

dataframe (*sportsreference.ncaaf.teams.Team* attribute), 90

dataframe (*sportsreference.nfl.boxscore.Boxscore* attribute), 95

dataframe (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute), 98

dataframe (*sportsreference.nfl.roster.Player* attribute), 104

dataframe (*sportsreference.nfl.schedule.Game* attribute), 110



- dataframe (*sportsreference.nfl.schedule.Schedule* attribute), 112
- dataframe (*sportsreference.nfl.teams.Team* attribute), 113
- dataframe (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- dataframe (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119
- dataframe (*sportsreference.nhl.roster.Player* attribute), 124
- dataframe (*sportsreference.nhl.schedule.Game* attribute), 128
- dataframe (*sportsreference.nhl.schedule.Schedule* attribute), 130
- dataframe (*sportsreference.nhl.teams.Team* attribute), 131
- dataframe\_extended (*sportsreference.mlb.schedule.Game* attribute), 20
- dataframe\_extended (*sportsreference.mlb.schedule.Schedule* attribute), 22
- dataframe\_extended (*sportsreference.nba.schedule.Game* attribute), 44
- dataframe\_extended (*sportsreference.nba.schedule.Schedule* attribute), 45
- dataframe\_extended (*sportsreference.ncaab.schedule.Game* attribute), 65
- dataframe\_extended (*sportsreference.ncaab.schedule.Schedule* attribute), 66
- dataframe\_extended (*sportsreference.ncaaf.schedule.Game* attribute), 88
- dataframe\_extended (*sportsreference.ncaaf.schedule.Schedule* attribute), 89
- dataframe\_extended (*sportsreference.nfl.schedule.Game* attribute), 110
- dataframe\_extended (*sportsreference.nfl.schedule.Schedule* attribute), 112
- dataframe\_extended (*sportsreference.nhl.schedule.Game* attribute), 128
- dataframe\_extended (*sportsreference.nhl.schedule.Schedule* attribute), 130
- dataframes (*sportsreference.mlb.teams.Teams* attribute), 28
- dataframes (*sportsreference.nba.teams.Teams* attribute), 49
- dataframes (*sportsreference.ncaab.teams.Teams* attribute), 72
- dataframes (*sportsreference.ncaaf.teams.Teams* attribute), 93
- dataframes (*sportsreference.nfl.teams.Teams* attribute), 115
- dataframes (*sportsreference.nhl.teams.Teams* attribute), 133
- date (*sportsreference.mlb.boxscore.Boxscore* attribute), 7
- date (*sportsreference.mlb.schedule.Game* attribute), 20
- date (*sportsreference.nba.boxscore.Boxscore* attribute), 31
- date (*sportsreference.nba.schedule.Game* attribute), 44
- date (*sportsreference.ncaab.boxscore.Boxscore* attribute), 52
- date (*sportsreference.ncaab.schedule.Game* attribute), 65
- date (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 74
- date (*sportsreference.ncaaf.schedule.Game* attribute), 88
- date (*sportsreference.nfl.boxscore.Boxscore* attribute), 95
- date (*sportsreference.nfl.schedule.Game* attribute), 110
- date (*sportsreference.nhl.boxscore.Boxscore* attribute), 117
- date (*sportsreference.nhl.schedule.Game* attribute), 128
- datetime (*sportsreference.mlb.schedule.Game* attribute), 20
- datetime (*sportsreference.nba.schedule.Game* attribute), 44
- datetime (*sportsreference.ncaab.schedule.Game* attribute), 65
- datetime (*sportsreference.ncaaf.schedule.Game* attribute), 88
- datetime (*sportsreference.nfl.schedule.Game* attribute), 110
- datetime (*sportsreference.nhl.schedule.Game* attribute), 128
- day (*sportsreference.nfl.schedule.Game* attribute), 110
- day\_of\_week (*sportsreference.ncaaf.schedule.Game* attribute), 88
- day\_or\_night (*sportsreference.mlb.schedule.Game* attribute), 20
- decision (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119
- defensive\_box\_plus\_minus (*sportsreference.nba.roster.Player* attribute), 40
- defensive\_box\_plus\_minus (*sportsreference.ncaab.roster.Player* attribute), 63
- defensive\_chances (*sportsreference.mlb.roster.Player* attribute), 15
- defensive\_point\_shares (*sportsreference.nhl.roster.Player* attribute), 124
- defensive\_rating (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 34
- defensive\_rating (*sportsreference.ncaab.boxscore.BoxscorePlayer* attribute), 52

- tribute), 55
- defensive\_rebound\_percentage (*sportsreference.nba.player.AbstractPlayer attribute*), 36
- defensive\_rebound\_percentage (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58
- defensive\_rebounds (*sportsreference.nba.player.AbstractPlayer attribute*), 36
- defensive\_rebounds (*sportsreference.nba.teams.Team attribute*), 46
- defensive\_rebounds (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58
- defensive\_rebounds (*sportsreference.ncaab.teams.Team attribute*), 68
- defensive\_runs\_saved\_above\_average (*sportsreference.mlb.roster.Player attribute*), 15
- defensive\_runs\_saved\_above\_average\_per\_inning (*sportsreference.mlb.roster.Player attribute*), 15
- defensive\_simple\_rating\_system (*sportsreference.nfl.teams.Team attribute*), 113
- defensive\_win\_shares (*sportsreference.nba.roster.Player attribute*), 40
- defensive\_win\_shares (*sportsreference.ncaab.roster.Player attribute*), 63
- defensive\_zone\_start\_percentage (*sportsreference.nhl.boxscore.BoxscorePlayer attribute*), 119
- defensive\_zone\_start\_percentage (*sportsreference.nhl.roster.Player attribute*), 124
- defensive\_zone\_starts (*sportsreference.nhl.boxscore.BoxscorePlayer attribute*), 119
- double\_plays\_turned (*sportsreference.mlb.roster.Player attribute*), 15
- doubles (*sportsreference.mlb.roster.Player attribute*), 15
- doubles (*sportsreference.mlb.teams.Team attribute*), 23
- dunks (*sportsreference.nba.roster.Player attribute*), 40
- duration (*sportsreference.mlb.boxscore.Boxscore attribute*), 7
- duration (*sportsreference.nfl.boxscore.Boxscore attribute*), 96
- duration (*sportsreference.nhl.boxscore.Boxscore attribute*), 117
- E**
- earned\_runs\_against (*sportsreference.mlb.boxscore.BoxscorePlayer attribute*), 10
- earned\_runs\_against (*sportsreference.mlb.teams.Team attribute*), 24
- earned\_runs\_against\_plus (*sportsreference.mlb.teams.Team attribute*), 24
- earned\_runs\_allowed (*sportsreference.mlb.player.AbstractPlayer attribute*), 13
- earned\_runs\_allowed (*sportsreference.mlb.roster.Player attribute*), 16
- effective\_field\_goal\_percentage (*sportsreference.nba.player.AbstractPlayer attribute*), 36
- effective\_field\_goal\_percentage (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58
- effective\_field\_goal\_percentage (*sportsreference.ncaab.teams.Team attribute*), 68
- era (*sportsreference.mlb.roster.Player attribute*), 16
- era\_plus (*sportsreference.mlb.roster.Player attribute*), 16
- errors (*sportsreference.mlb.roster.Player attribute*), 16
- errors\_per\_inning (*sportsreference.nfl.roster.Player attribute*), 104
- even\_strength\_assists (*sportsreference.nhl.player.AbstractPlayer attribute*), 121
- even\_strength\_goals (*sportsreference.nhl.player.AbstractPlayer attribute*), 121
- even\_strength\_goals\_allowed (*sportsreference.nhl.roster.Player attribute*), 124
- even\_strength\_save\_percentage (*sportsreference.nhl.roster.Player attribute*), 124
- even\_strength\_shots\_faced (*sportsreference.nhl.roster.Player attribute*), 124
- extra\_inning\_losses (*sportsreference.mlb.teams.Team attribute*), 24
- extra\_inning\_record (*sportsreference.mlb.teams.Team attribute*), 24
- extra\_inning\_wins (*sportsreference.mlb.teams.Team attribute*), 24
- extra\_point\_percentage (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 76
- extra\_point\_percentage (*sportsreference.nfl.roster.Player attribute*), 104
- extra\_points\_attempted (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 76
- extra\_points\_attempted (*sportsreference.nfl.player.AbstractPlayer attribute*), 100
- extra\_points\_attempted (*sportsreference.nfl.roster.Player attribute*), 104
- extra\_points\_attempted (*sportsreference*

- ence.nfl.schedule.Game attribute*), 110  
 extra\_points\_made (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 80  
 extra\_points\_made (*sportsreference.ncaaf.roster.Player attribute*), 84  
 extra\_points\_made (*sportsreference.nfl.player.AbstractPlayer attribute*), 100  
 extra\_points\_made (*sportsreference.nfl.roster.Player attribute*), 104  
 extra\_points\_made (*sportsreference.nfl.schedule.Game attribute*), 110
- ## F
- faceoff\_losses (*sportsreference.nhl.roster.Player attribute*), 124  
 faceoff\_losses (*sportsreference.nhl.schedule.Game attribute*), 128  
 faceoff\_percentage (*sportsreference.nhl.roster.Player attribute*), 124  
 faceoff\_win\_percentage (*sportsreference.nhl.schedule.Game attribute*), 128  
 faceoff\_wins (*sportsreference.nhl.roster.Player attribute*), 124  
 faceoff\_wins (*sportsreference.nhl.schedule.Game attribute*), 129  
 fenwick\_against (*sportsreference.nhl.roster.Player attribute*), 124  
 fenwick\_against (*sportsreference.nhl.schedule.Game attribute*), 129  
 fenwick\_for (*sportsreference.nhl.roster.Player attribute*), 124  
 fenwick\_for (*sportsreference.nhl.schedule.Game attribute*), 129  
 fenwick\_for\_percentage (*sportsreference.nhl.roster.Player attribute*), 124  
 fenwick\_for\_percentage (*sportsreference.nhl.schedule.Game attribute*), 129  
 field\_goal\_attempts (*sportsreference.nba.player.AbstractPlayer attribute*), 37  
 field\_goal\_attempts (*sportsreference.nba.teams.Team attribute*), 46  
 field\_goal\_attempts (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58  
 field\_goal\_attempts (*sportsreference.ncaab.teams.Team attribute*), 68  
 field\_goal\_perc\_sixteen\_foot\_plus\_two\_points (*sportsreference.nba.roster.Player attribute*), 40  
 field\_goal\_perc\_ten\_to\_sixteen\_feet (*sportsreference.nba.roster.Player attribute*), 40  
 field\_goal\_perc\_three\_to\_ten\_feet (*sportsreference.nba.roster.Player attribute*), 40  
 field\_goal\_perc\_zero\_to\_three\_feet (*sportsreference.nba.roster.Player attribute*), 40  
 field\_goal\_percentage (*sportsreference.nba.player.AbstractPlayer attribute*), 37  
 field\_goal\_percentage (*sportsreference.nba.teams.Team attribute*), 46  
 field\_goal\_percentage (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58  
 field\_goal\_percentage (*sportsreference.ncaab.teams.Team attribute*), 68  
 field\_goal\_percentage (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 76  
 field\_goal\_percentage (*sportsreference.nfl.roster.Player attribute*), 104  
 field\_goals (*sportsreference.nba.player.AbstractPlayer attribute*), 37  
 field\_goals (*sportsreference.nba.teams.Team attribute*), 46  
 field\_goals (*sportsreference.ncaab.player.AbstractPlayer attribute*), 58  
 field\_goals (*sportsreference.ncaab.teams.Team attribute*), 68  
 field\_goals\_attempted (*sportsreference.ncaaf.boxscore.BoxscorePlayer attribute*), 76  
 field\_goals\_attempted (*sportsreference.nfl.player.AbstractPlayer attribute*), 100  
 field\_goals\_attempted (*sportsreference.nfl.roster.Player attribute*), 104  
 field\_goals\_attempted (*sportsreference.nfl.schedule.Game attribute*), 110  
 field\_goals\_made (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 80  
 field\_goals\_made (*sportsreference.ncaaf.roster.Player attribute*), 84  
 field\_goals\_made (*sportsreference.nfl.player.AbstractPlayer attribute*), 100  
 field\_goals\_made (*sportsreference.nfl.roster.Player attribute*), 104  
 field\_goals\_made (*sportsreference.nfl.schedule.Game attribute*), 110  
 fielding\_independent\_pitching (*sportsrefer-*

*ence.mlb.roster.Player attribute*), 16

fielding\_independent\_pitching (*sportsreference.mlb.teams.Team attribute*), 24

fielding\_percentage (*sportsreference.mlb.roster.Player attribute*), 16

fifty\_plus\_yard\_field\_goal\_attempts (*sportsreference.nfl.roster.Player attribute*), 104

fifty\_plus\_yard\_field\_goals\_made (*sportsreference.nfl.roster.Player attribute*), 104

first\_downs (*sportsreference.ncaaf.teams.Team attribute*), 90

first\_downs (*sportsreference.nfl.teams.Team attribute*), 113

first\_downs\_from\_penalties (*sportsreference.ncaaf.teams.Team attribute*), 90

first\_downs\_from\_penalties (*sportsreference.nfl.teams.Team attribute*), 113

fly\_balls (*sportsreference.mlb.boxscore.BoxscorePlayer attribute*), 10

fourth\_down\_attempts (*sportsreference.nfl.schedule.Game attribute*), 110

fourth\_down\_conversions (*sportsreference.nfl.schedule.Game attribute*), 110

fourth\_quarter\_comebacks (*sportsreference.nfl.roster.Player attribute*), 104

fourty\_to\_fourty\_nine\_yard\_field\_goal\_attempts (*sportsreference.nfl.roster.Player attribute*), 104

fourty\_to\_fourty\_nine\_yard\_field\_goals\_made (*sportsreference.nfl.roster.Player attribute*), 104

free\_throw\_attempt\_rate (*sportsreference.nba.player.AbstractPlayer attribute*), 37

free\_throw\_attempt\_rate (*sportsreference.ncaab.player.AbstractPlayer attribute*), 59

free\_throw\_attempt\_rate (*sportsreference.ncaab.teams.Team attribute*), 68

free\_throw\_attempts (*sportsreference.nba.player.AbstractPlayer attribute*), 37

free\_throw\_attempts (*sportsreference.nba.teams.Team attribute*), 46

free\_throw\_attempts (*sportsreference.ncaab.player.AbstractPlayer attribute*), 59

free\_throw\_attempts (*sportsreference.ncaab.teams.Team attribute*), 68

free\_throw\_percentage (*sportsreference.nba.teams.Team attribute*), 46

free\_throw\_percentage (*sportsreference.ncaab.player.AbstractPlayer attribute*), 59

free\_throw\_percentage (*sportsreference.ncaab.teams.Team attribute*), 68

free\_throws (*sportsreference.nba.player.AbstractPlayer attribute*), 37

free\_throws (*sportsreference.nba.teams.Team attribute*), 46

free\_throws (*sportsreference.ncaab.player.AbstractPlayer attribute*), 59

free\_throws (*sportsreference.ncaab.teams.Team attribute*), 68

free\_throws\_per\_field\_goal\_attempt (*sportsreference.ncaab.teams.Team attribute*), 68

fumbles (*sportsreference.nfl.player.AbstractPlayer attribute*), 100

fumbles (*sportsreference.nfl.roster.Player attribute*), 104

fumbles (*sportsreference.nfl.teams.Team attribute*), 113

fumbles\_forced (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 85

fumbles\_forced (*sportsreference.ncaaf.roster.Player attribute*), 85

fumbles\_forced (*sportsreference.nfl.player.AbstractPlayer attribute*), 100

fumbles\_forced (*sportsreference.nfl.roster.Player attribute*), 105

fumbles\_lost (*sportsreference.ncaaf.teams.Team attribute*), 90

fumbles\_lost (*sportsreference.nfl.boxscore.BoxscorePlayer attribute*), 98

fumbles\_recovered (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 80

fumbles\_recovered (*sportsreference.ncaaf.roster.Player attribute*), 85

fumbles\_recovered (*sportsreference.nfl.player.AbstractPlayer attribute*), 100

fumbles\_recovered (*sportsreference.nfl.roster.Player attribute*), 105

fumbles\_recovered\_for\_touchdown (*sportsreference.ncaaf.player.AbstractPlayer attribute*), 80

fumbles\_recovered\_for\_touchdown (*sportsref-*

*erence.ncaaf.roster.Player attribute*), 85  
*fumbles\_recovered\_for\_touchdown (sportsreference.nfl.player.AbstractPlayer attribute)*, 100  
*fumbles\_recovered\_for\_touchdown (sportsreference.nfl.roster.Player attribute)*, 105

## G

*Game (class in sportsreference.mlb.schedule)*, 20  
*Game (class in sportsreference.nba.schedule)*, 43  
*Game (class in sportsreference.ncaab.schedule)*, 64  
*Game (class in sportsreference.ncaaf.schedule)*, 87  
*Game (class in sportsreference.nfl.schedule)*, 109  
*Game (class in sportsreference.nhl.schedule)*, 128  
*game (sportsreference.mlb.schedule.Game attribute)*, 21  
*game (sportsreference.nba.schedule.Game attribute)*, 44  
*game (sportsreference.ncaab.schedule.Game attribute)*, 65  
*game (sportsreference.ncaaf.schedule.Game attribute)*, 88  
*game (sportsreference.nhl.schedule.Game attribute)*, 129  
*game\_duration (sportsreference.mlb.schedule.Game attribute)*, 21  
*game\_number\_for\_day (sportsreference.mlb.schedule.Game attribute)*, 21  
*game\_score (sportsreference.mlb.boxscore.BoxscorePlayer attribute)*, 10  
*game\_winning\_drives (sportsreference.nfl.roster.Player attribute)*, 105  
*game\_winning\_goals (sportsreference.nhl.player.AbstractPlayer attribute)*, 121  
*games (sportsreference.mlb.boxscore.Boxscores attribute)*, 11  
*games (sportsreference.mlb.roster.Player attribute)*, 16  
*games (sportsreference.mlb.teams.Team attribute)*, 24  
*games (sportsreference.nba.boxscore.Boxscores attribute)*, 35  
*games (sportsreference.ncaab.boxscore.Boxscores attribute)*, 55  
*games (sportsreference.ncaaf.boxscore.Boxscores attribute)*, 77  
*games (sportsreference.ncaaf.roster.Player attribute)*, 85  
*games (sportsreference.ncaaf.teams.Team attribute)*, 90  
*games (sportsreference.nfl.boxscore.Boxscores attribute)*, 98  
*games (sportsreference.nfl.roster.Player attribute)*, 105  
*games (sportsreference.nhl.boxscore.Boxscores attribute)*, 120  
*games\_behind (sportsreference.mlb.schedule.Game attribute)*, 21  
*games\_catcher (sportsreference.mlb.roster.Player attribute)*, 16

*games\_center\_fielder (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_designated\_hitter (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_finished (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_finished (sportsreference.mlb.teams.Team attribute)*, 24  
*games\_first\_baseman (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_in\_batting\_order (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_in\_defensive\_lineup (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_left\_fielder (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_outfielder (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_pinch\_hitter (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_pinch\_runner (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_pitcher (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_played (sportsreference.nba.roster.Player attribute)*, 40  
*games\_played (sportsreference.nba.teams.Team attribute)*, 46  
*games\_played (sportsreference.ncaab.roster.Player attribute)*, 63  
*games\_played (sportsreference.ncaab.teams.Team attribute)*, 68  
*games\_played (sportsreference.nfl.teams.Team attribute)*, 113  
*games\_played (sportsreference.nhl.roster.Player attribute)*, 125  
*games\_played (sportsreference.nhl.teams.Team attribute)*, 131  
*games\_right\_fielder (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_second\_baseman (sportsreference.mlb.roster.Player attribute)*, 16  
*games\_shortstop (sportsreference.mlb.roster.Player attribute)*, 17  
*games\_started (sportsreference.mlb.roster.Player attribute)*, 17  
*games\_started (sportsreference.nba.roster.Player attribute)*, 40  
*games\_started (sportsreference.ncaab.roster.Player attribute)*, 63  
*games\_started (sportsreference.nfl.roster.Player attribute)*, 105  
*games\_third\_baseman (sportsreference.mlb.roster.Player attribute)*, 17



giveaways (*sportsreference.nhl.roster.Player attribute*), 125

goal\_against\_percentage\_relative (*sportsreference.nhl.roster.Player attribute*), 125

goalie\_point\_shares (*sportsreference.nhl.roster.Player attribute*), 125

goals (*sportsreference.nhl.player.AbstractPlayer attribute*), 121

goals\_against (*sportsreference.nhl.player.AbstractPlayer attribute*), 121

goals\_against (*sportsreference.nhl.teams.Team attribute*), 131

goals\_against\_average (*sportsreference.nhl.roster.Player attribute*), 125

goals\_against\_on\_ice (*sportsreference.nhl.roster.Player attribute*), 125

goals\_allowed (*sportsreference.nhl.schedule.Game attribute*), 129

goals\_created (*sportsreference.nhl.roster.Player attribute*), 125

goals\_for (*sportsreference.nhl.teams.Team attribute*), 131

goals\_for\_on\_ice (*sportsreference.nhl.roster.Player attribute*), 125

goals\_saved\_above\_average (*sportsreference.nhl.roster.Player attribute*), 125

goals\_scored (*sportsreference.nhl.schedule.Game attribute*), 129

grounded\_balls (*sportsreference.mlb.boxscore.BoxscorePlayer attribute*), 10

grounded\_into\_double\_plays (*sportsreference.mlb.roster.Player attribute*), 17

grounded\_into\_double\_plays (*sportsreference.mlb.teams.Team attribute*), 24

## H

half\_court\_heaves (*sportsreference.nba.roster.Player attribute*), 40

half\_court\_heaves\_made (*sportsreference.nba.roster.Player attribute*), 40

height (*sportsreference.mlb.roster.Player attribute*), 17

height (*sportsreference.nba.roster.Player attribute*), 40

height (*sportsreference.ncaab.roster.Player attribute*), 63

height (*sportsreference.ncaaf.roster.Player attribute*), 85

height (*sportsreference.nfl.roster.Player attribute*), 105

height (*sportsreference.nhl.roster.Player attribute*), 125

hit\_pitcher (*sportsreference.mlb.teams.Team attribute*), 24

hits (*sportsreference.mlb.player.AbstractPlayer attribute*), 13

hits (*sportsreference.mlb.roster.Player attribute*), 17

hits (*sportsreference.mlb.teams.Team attribute*), 24

hits\_against\_per\_nine\_innings (*sportsreference.mlb.roster.Player attribute*), 17

hits\_allowed (*sportsreference.mlb.player.AbstractPlayer attribute*), 13

hits\_allowed (*sportsreference.mlb.roster.Player attribute*), 17

hits\_allowed (*sportsreference.mlb.teams.Team attribute*), 24

hits\_at\_even\_strength (*sportsreference.nhl.player.AbstractPlayer attribute*), 121

hits\_per\_nine\_innings (*sportsreference.mlb.teams.Team attribute*), 24

home\_abbreviation (*sportsreference.nfl.boxscore.Boxscore attribute*), 96

home\_assist\_percentage (*sportsreference.nba.boxscore.Boxscore attribute*), 31

home\_assist\_percentage (*sportsreference.ncaab.boxscore.Boxscore attribute*), 52

home\_assists (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_assists (*sportsreference.nba.boxscore.Boxscore attribute*), 31

home\_assists (*sportsreference.ncaab.boxscore.Boxscore attribute*), 52

home\_assists (*sportsreference.nhl.boxscore.Boxscore attribute*), 117

home\_at\_bats (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_average\_leverage\_index (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_base\_out\_runs\_added (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_base\_out\_runs\_saved (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_bases\_on\_balls (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_batting\_average (*sportsreference.mlb.boxscore.Boxscore attribute*), 7

home\_block\_percentage (*sportsreference.nba.boxscore.Boxscore attribute*), 31

home\_block\_percentage (*sportsreference.ncaab.boxscore.Boxscore attribute*), 52

home\_blocks (*sportsreference.nba.boxscore.Boxscore attribute*), 31

home\_blocks (*sportsreference*

<i>ence.ncaab.boxscore.Boxscore</i>	<i>attribute</i> ),	<i>ence.nfl.boxscore.Boxscore</i>	<i>attribute</i> ), 96
52		home_free_throw_attempt_rate	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32
home_defensive_rating	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_free_throw_attempt_rate	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),
home_defensive_rating	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	52	
52		home_free_throw_attempts	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32
home_defensive_rebound_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_free_throw_attempts	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),
home_defensive_rebound_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	53	
52		home_free_throw_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32
home_defensive_rebounds	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_free_throw_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),
home_defensive_rebounds	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	53	
52		home_free_throws	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32
home_earned_runs	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 7	home_free_throws	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),
home_effective_field_goal_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	53	
home_effective_field_goal_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ), 52	home_fumbles	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> <i>attribute</i> ),
		74	
home_even_strength_assists	( <i>sportsreference.nhl.boxscore.Boxscore</i> <i>attribute</i> ), 117	home_fumbles	( <i>sportsreference.nfl.boxscore.Boxscore</i> <i>attribute</i> ), 96
home_even_strength_goals	( <i>sportsreference.nhl.boxscore.Boxscore</i> <i>attribute</i> ), 117	home_fumbles_lost	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> <i>attribute</i> ),
		74	
home_field_goal_attempts	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_fumbles_lost	( <i>sportsreference.nfl.boxscore.Boxscore</i> <i>attribute</i> ), 96
home_field_goal_attempts	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	home_game_score	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 7
52		home_game_winning_goals	( <i>sportsreference.nhl.boxscore.Boxscore</i> <i>attribute</i> ), 117
home_field_goal_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_goals	( <i>sportsreference.nhl.boxscore.Boxscore</i> <i>attribute</i> ), 118
home_field_goal_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	home_grounded_balls	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
52		home_hits	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
home_field_goals	( <i>sportsreference.nba.boxscore.Boxscore</i> <i>attribute</i> ), 32	home_home_runs	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
home_field_goals	( <i>sportsreference.ncaab.boxscore.Boxscore</i> <i>attribute</i> ),	home_inherited_runners	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
52		home_inherited_score	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
home_first_downs	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> <i>attribute</i> ),	home_innings_pitched	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 8
74		home_interceptions	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> <i>attribute</i> ),
home_first_downs	( <i>sportsreference.nfl.boxscore.Boxscore</i> <i>attribute</i> ), 96	74	
home_fly_balls	( <i>sportsreference.mlb.boxscore.Boxscore</i> <i>attribute</i> ), 7	home_interceptions	( <i>sportsreference.nfl.boxscore.Boxscore</i> <i>attribute</i> ), 96
home_fourth_down_attempts	( <i>sportsreference.nfl.boxscore.Boxscore</i> <i>attribute</i> ), 96		
home_fourth_down_conversions	( <i>sportsrefer-</i>		

home\_line\_drives (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_losses (sportsreference.mlb.teams.Team attribute), 24

home\_losses (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_losses (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_losses (sportsreference.ncaab.teams.Team attribute), 68

home\_minutes\_played (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_minutes\_played (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_net\_pass\_yards (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_offensive\_rating (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_offensive\_rating (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_offensive\_rebound\_percentage (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_offensive\_rebound\_percentage (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_offensive\_rebounds (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_offensive\_rebounds (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_on\_base\_percentage (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_on\_base\_plus (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_pass\_attempts (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_pass\_attempts (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_pass\_completions (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_pass\_completions (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_pass\_touchdowns (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_pass\_touchdowns (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_pass\_yards (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_pass\_yards (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_penalties (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_penalties (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_penalties\_in\_minutes (sportsreference.nhl.boxscore.Boxscore attribute), 118

home\_personal\_fouls (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_personal\_fouls (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_pitches (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_plate\_appearances (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_players (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_players (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_players (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_players (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_players (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_players (sportsreference.nhl.boxscore.Boxscore attribute), 118

home\_points (sportsreference.nba.boxscore.Boxscore attribute), 32

home\_points (sportsreference.ncaab.boxscore.Boxscore attribute), 53

home\_points (sportsreference.ncaaf.boxscore.Boxscore attribute), 74

home\_points (sportsreference.nfl.boxscore.Boxscore attribute), 96

home\_points (sportsreference.nhl.boxscore.Boxscore attribute), 118

home\_power\_play\_assists (sportsreference.nhl.boxscore.Boxscore attribute), 118

home\_power\_play\_goals (sportsreference.nhl.boxscore.Boxscore attribute), 118

home\_putouts (sportsreference.mlb.boxscore.Boxscore attribute), 8

home\_ranking (sportsreference.ncaab.boxscore.Boxscore attribute), 53



home_rbi ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8	ence.nba.boxscore.Boxscore attribute), 33
home_record ( <i>sportsreference.mlb.teams.Team attribute</i> ), 24	home_steal_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_runs ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8	home_steals ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_runs ( <i>sportsreference.mlb.roster.Player attribute</i> ), 17	home_steals ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_runs ( <i>sportsreference.mlb.teams.Team attribute</i> ), 24	home_strikeouts ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8
home_runs_against ( <i>sportsreference.mlb.teams.Team attribute</i> ), 24	home_strikes ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8
home_runs_against_per_nine_innings ( <i>sportsreference.mlb.roster.Player attribute</i> ), 17	home_strikes_by_contact ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8
home_runs_allowed ( <i>sportsreference.mlb.roster.Player attribute</i> ), 17	home_strikes_looking ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8
home_runs_per_nine_innings ( <i>sportsreference.mlb.teams.Team attribute</i> ), 24	home_strikes_swinging ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8
home_runs_thrown ( <i>sportsreference.mlb.boxscore.BoxscorePlayer attribute</i> ), 10	home_third_down_attempts ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 96
home_rush_attempts ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 74	home_third_down_conversions ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 96
home_rush_attempts ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 96	home_three_point_attempt_rate ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_rush_touchdowns ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 74	home_three_point_attempt_rate ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_rush_touchdowns ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 96	home_three_point_field_goal_attempts ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_rush_yards ( <i>sportsreference.ncaaf.boxscore.Boxscore attribute</i> ), 74	home_three_point_field_goal_attempts ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_rush_yards ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 96	home_three_point_field_goal_percentage ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_save_percentage ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_three_point_field_goal_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_saves ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_three_point_field_goals ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_shooting_percentage ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_three_point_field_goals ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_short_handed_assists ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_time_of_possession ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 97
home_short_handed_goals ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_times_sacked ( <i>sportsreference.nfl.boxscore.Boxscore attribute</i> ), 97
home_shots_on_goal ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_total_rebound_percentage ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_shutout ( <i>sportsreference.nhl.boxscore.Boxscore attribute</i> ), 118	home_total_rebound_percentage ( <i>sportsreference.ncaab.boxscore.Boxscore attribute</i> ), 53
home_slugging_percentage ( <i>sportsreference.mlb.boxscore.Boxscore attribute</i> ), 8	home_total_rebounds ( <i>sportsreference.nba.boxscore.Boxscore attribute</i> ), 33
home_steal_percentage ( <i>sportsrefer-</i>	

home_total_rebounds	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	( <i>sportsreference.mlb.boxscore.Boxscore</i> attribute), 9
home_total_yards	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> attribute), 74	home_win_probability_subtracted ( <i>sportsreference.mlb.boxscore.Boxscore</i> attribute), 9
home_total_yards	( <i>sportsreference.nfl.boxscore.Boxscore</i> attribute), 97	home_wins ( <i>sportsreference.mlb.teams.Team</i> attribute), 24
home_true_shooting_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	home_wins ( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33
home_true_shooting_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	home_wins ( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54
home_turnover_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	home_wins ( <i>sportsreference.ncaab.teams.Team</i> attribute), 68
home_turnover_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	home_yards_from_penalties ( <i>sportsreference.ncaaf.boxscore.Boxscore</i> attribute), 75
home_turnovers	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	home_yards_from_penalties ( <i>sportsreference.nfl.boxscore.Boxscore</i> attribute), 97
home_turnovers	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	home_yards_lost_from_sacks ( <i>sportsreference.nfl.boxscore.Boxscore</i> attribute), 97
home_turnovers	( <i>sportsreference.ncaaf.boxscore.Boxscore</i> attribute), 74	
home_turnovers	( <i>sportsreference.nfl.boxscore.Boxscore</i> attribute), 97	
home_two_point_field_goal_attempts	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	individual_corsi_for_events ( <i>sportsreference.nhl.boxscore.BoxscorePlayer</i> attribute), 119
home_two_point_field_goal_attempts	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	inherited_runners ( <i>sportsreference.mlb.boxscore.BoxscorePlayer</i> attribute), 10
home_two_point_field_goal_percentage	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	inherited_score ( <i>sportsreference.mlb.boxscore.BoxscorePlayer</i> attribute), 10
home_two_point_field_goal_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	innings ( <i>sportsreference.mlb.schedule.Game</i> attribute), 21
home_two_point_field_goals	( <i>sportsreference.nba.boxscore.Boxscore</i> attribute), 33	innings_pitched ( <i>sportsreference.mlb.boxscore.BoxscorePlayer</i> attribute), 10
home_two_point_field_goals	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	innings_pitched ( <i>sportsreference.mlb.teams.Team</i> attribute), 24
home_unknown_bat_type	( <i>sportsreference.mlb.boxscore.Boxscore</i> attribute), 9	innings_played ( <i>sportsreference.mlb.roster.Player</i> attribute), 17
home_win_percentage	( <i>sportsreference.ncaab.boxscore.Boxscore</i> attribute), 54	intentional_bases_on_balls ( <i>sportsreference.mlb.roster.Player</i> attribute), 17
home_win_probability_added	( <i>sportsreference.mlb.boxscore.Boxscore</i> attribute), 9	intentional_bases_on_balls ( <i>sportsreference.mlb.teams.Team</i> attribute), 25
home_win_probability_by_pitcher	( <i>sportsreference.mlb.boxscore.Boxscore</i> attribute), 9	intentional_bases_on_balls_given ( <i>sportsreference.mlb.roster.Player</i> attribute), 17
home_win_probability_for_offensive_player		interception_percentage ( <i>sportsreference.nfl.roster.Player</i> attribute), 105
		interception_percentage_index ( <i>sportsreference.nfl.roster.Player</i> attribute), 105
		interceptions ( <i>sportsreference.ncaaf.player.AbstractPlayer</i> attribute), 80
		interceptions ( <i>sportsreference.ncaaf.roster.Player</i> attribute), 85

- [interceptions](#) (*sportsreference.ncaafootball.teams.Team* attribute), 91  
[interceptions](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[interceptions](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[interceptions](#) (*sportsreference.nfl.schedule.Game* attribute), 110  
[interceptions](#) (*sportsreference.nfl.teams.Team* attribute), 113  
[interceptions\\_returned\\_for\\_touchdown](#) (*sportsreference.ncaafootball.player.AbstractPlayer* attribute), 80  
[interceptions\\_returned\\_for\\_touchdown](#) (*sportsreference.ncaafootball.roster.Player* attribute), 85  
[interceptions\\_returned\\_for\\_touchdown](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[interceptions\\_returned\\_for\\_touchdown](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[interceptions\\_thrown](#) (*sportsreference.ncaafootball.player.AbstractPlayer* attribute), 80  
[interceptions\\_thrown](#) (*sportsreference.ncaafootball.roster.Player* attribute), 85  
[interceptions\\_thrown](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[interceptions\\_thrown](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[interleague\\_record](#) (*sportsreference.mlb.teams.Team* attribute), 25
- ## K
- [kickoff\\_return\\_touchdown](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[kickoff\\_return\\_touchdown](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[kickoff\\_return\\_touchdowns](#) (*sportsreference.ncaafootball.player.AbstractPlayer* attribute), 80  
[kickoff\\_return\\_touchdowns](#) (*sportsreference.ncaafootball.roster.Player* attribute), 85  
[kickoff\\_return\\_yards](#) (*sportsreference.ncaafootball.boxscore.BoxscorePlayer* attribute), 76  
[kickoff\\_return\\_yards](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[kickoff\\_return\\_yards](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[kickoff\\_returns](#) (*sportsreference.ncaafootball.boxscore.BoxscorePlayer* attribute), 76  
[kickoff\\_returns](#) (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
[kickoff\\_returns](#) (*sportsreference.nfl.roster.Player* attribute), 105
- ## L
- [last\\_ten\\_games\\_record](#) (*sportsreference.mlb.teams.Team* attribute), 25  
[last\\_thirty\\_games\\_record](#) (*sportsreference.mlb.teams.Team* attribute), 25  
[last\\_twenty\\_games\\_record](#) (*sportsreference.mlb.teams.Team* attribute), 25  
[league](#) (*sportsreference.mlb.teams.Team* attribute), 25  
[league](#) (*sportsreference.nhl.roster.Player* attribute), 125  
[league\\_fielding\\_percentage](#) (*sportsreference.mlb.roster.Player* attribute), 17  
[league\\_range\\_factor\\_per\\_game](#) (*sportsreference.mlb.roster.Player* attribute), 17  
[league\\_range\\_factor\\_per\\_nine\\_innings](#) (*sportsreference.mlb.roster.Player* attribute), 17  
[less\\_than\\_nineteen\\_yards\\_field\\_goal\\_attempts](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[less\\_than\\_nineteen\\_yards\\_field\\_goals\\_made](#) (*sportsreference.nfl.roster.Player* attribute), 105  
[line\\_drives](#) (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 10  
[location](#) (*sportsreference.mlb.schedule.Game* attribute), 21  
[location](#) (*sportsreference.nba.boxscore.Boxscore* attribute), 33  
[location](#) (*sportsreference.nba.schedule.Game* attribute), 44  
[location](#) (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
[location](#) (*sportsreference.ncaab.schedule.Game* attribute), 65  
[location](#) (*sportsreference.ncaafootball.schedule.Game* attribute), 88  
[location](#) (*sportsreference.nfl.schedule.Game* attribute), 110  
[location](#) (*sportsreference.nhl.schedule.Game* attribute), 129

`longest_field_goal_made` (*sportsreference.nfl.roster.Player* attribute), 105  
`longest_interception_return` (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
`longest_interception_return` (*sportsreference.nfl.roster.Player* attribute), 105  
`longest_kickoff_return` (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
`longest_kickoff_return` (*sportsreference.nfl.roster.Player* attribute), 105  
`longest_pass` (*sportsreference.nfl.player.AbstractPlayer* attribute), 100  
`longest_pass` (*sportsreference.nfl.roster.Player* attribute), 106  
`longest_punt` (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
`longest_punt` (*sportsreference.nfl.roster.Player* attribute), 106  
`longest_punt_return` (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
`longest_punt_return` (*sportsreference.nfl.roster.Player* attribute), 106  
`longest_reception` (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
`longest_reception` (*sportsreference.nfl.roster.Player* attribute), 106  
`longest_rush` (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
`longest_rush` (*sportsreference.nfl.roster.Player* attribute), 106  
`loser` (*sportsreference.mlb.schedule.Game* attribute), 21  
`losing_abbr` (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
`losing_abbr` (*sportsreference.nba.boxscore.Boxscore* attribute), 33  
`losing_abbr` (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
`losing_abbr` (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
`losing_abbr` (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
`losing_abbr` (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
`losing_name` (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
`losing_name` (*sportsreference.nba.boxscore.Boxscore* attribute), 33  
`losing_name` (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
`losing_name` (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
`losing_name` (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
`losing_name` (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
`losses` (*sportsreference.mlb.roster.Player* attribute), 17  
`losses` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses` (*sportsreference.nba.schedule.Game* attribute), 44  
`losses` (*sportsreference.ncaab.teams.Team* attribute), 68  
`losses` (*sportsreference.ncaaf.schedule.Game* attribute), 88  
`losses` (*sportsreference.ncaaf.teams.Team* attribute), 91  
`losses` (*sportsreference.nfl.teams.Team* attribute), 113  
`losses` (*sportsreference.nhl.roster.Player* attribute), 125  
`losses` (*sportsreference.nhl.teams.Team* attribute), 131  
`losses_last_ten_games` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_last_thirty_games` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_last_twenty_games` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_vs_left_handed_pitchers` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_vs_right_handed_pitchers` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_vs_teams_over_500` (*sportsreference.mlb.teams.Team* attribute), 25  
`losses_vs_teams_under_500` (*sportsreference.mlb.teams.Team* attribute), 25  
`lost_ball_turnovers` (*sportsreference.nba.roster.Player* attribute), 40  
`luck` (*sportsreference.mlb.teams.Team* attribute), 25

## M

`margin_of_victory` (*sportsreference.nfl.teams.Team* attribute), 113  
`minutes` (*sportsreference.nhl.roster.Player* attribute), 125  
`minutes_played` (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 34

minutes\_played (sportsreference.nba.player.AbstractPlayer attribute), 37

minutes\_played (sportsreference.nba.teams.Team attribute), 46

minutes\_played (sportsreference.ncaab.player.AbstractPlayer attribute), 59

minutes\_played (sportsreference.ncaab.teams.Team attribute), 68

mlb\_int\_property\_decorator() (in module sportsreference.mlb.teams), 28

## N

name (sportsreference.mlb.player.AbstractPlayer attribute), 13

name (sportsreference.mlb.roster.Player attribute), 17

name (sportsreference.mlb.teams.Team attribute), 25

name (sportsreference.nba.player.AbstractPlayer attribute), 37

name (sportsreference.nba.teams.Team attribute), 47

name (sportsreference.ncaab.player.AbstractPlayer attribute), 59

name (sportsreference.ncaab.teams.Team attribute), 68

name (sportsreference.ncaaf.player.AbstractPlayer attribute), 80

name (sportsreference.ncaaf.teams.Team attribute), 91

name (sportsreference.nfl.player.AbstractPlayer attribute), 101

name (sportsreference.nfl.teams.Team attribute), 113

name (sportsreference.nhl.player.AbstractPlayer attribute), 121

name (sportsreference.nhl.roster.Player attribute), 125

name (sportsreference.nhl.teams.Team attribute), 131

nationality (sportsreference.mlb.roster.Player attribute), 17

nationality (sportsreference.nba.roster.Player attribute), 40

ncaaf\_int\_property\_sub\_index() (in module sportsreference.ncaaf.boxscore), 77

net\_plus\_minus (sportsreference.nba.roster.Player attribute), 40

net\_rating (sportsreference.ncaab.teams.Team attribute), 68

net\_yards\_per\_attempt\_index (sportsreference.nfl.roster.Player attribute), 106

net\_yards\_per\_pass\_attempt (sportsreference.nfl.roster.Player attribute), 106

nfl\_int\_property\_sub\_index() (in module sportsreference.nfl.boxscore), 99

nhl\_int\_property\_decorator() (in module sportsreference.nhl.boxscore), 120

number\_of\_pitchers (sportsreference.mlb.teams.Team attribute), 25

number\_players\_used (sportsreference.mlb.teams.Team attribute), 25

## O

offensive\_box\_plus\_minus (sportsreference.nba.roster.Player attribute), 40

offensive\_box\_plus\_minus (sportsreference.ncaab.roster.Player attribute), 63

offensive\_fouls (sportsreference.nba.roster.Player attribute), 40

offensive\_point\_shares (sportsreference.nhl.roster.Player attribute), 125

offensive\_rating (sportsreference.nba.boxscore.BoxscorePlayer attribute), 34

offensive\_rating (sportsreference.ncaab.boxscore.BoxscorePlayer attribute), 55

offensive\_rating (sportsreference.ncaab.teams.Team attribute), 68

offensive\_rebound\_percentage (sportsreference.nba.player.AbstractPlayer attribute), 37

offensive\_rebound\_percentage (sportsreference.ncaab.player.AbstractPlayer attribute), 59

offensive\_rebound\_percentage (sportsreference.ncaab.teams.Team attribute), 69

offensive\_rebounds (sportsreference.nba.player.AbstractPlayer attribute), 37

offensive\_rebounds (sportsreference.nba.teams.Team attribute), 47

offensive\_rebounds (sportsreference.ncaab.player.AbstractPlayer attribute), 59

offensive\_rebounds (sportsreference.ncaab.teams.Team attribute), 69

offensive\_simple\_rating\_system (sportsreference.nfl.teams.Team attribute), 113

offensive\_win\_shares (sportsreference.nba.roster.Player attribute), 40

offensive\_win\_shares (sportsreference.ncaab.roster.Player attribute), 63

offensive\_zone\_start\_percentage (sportsreference.nhl.player.AbstractPlayer attribute), 121

offensive\_zone\_start\_percentage (sportsreference.nhl.schedule.Game attribute), 129

offensive\_zone\_starts (sportsreference.nhl.boxscore.BoxscorePlayer attribute), 119

on\_base\_percentage (sportsreference.mlb.player.AbstractPlayer attribute),



13  
on\_base\_percentage (*sportsreference.mlb.roster.Player* attribute), 18  
on\_base\_percentage (*sportsreference.mlb.teams.Team* attribute), 25  
on\_base\_plus\_slugging\_percentage (*sportsreference.mlb.player.AbstractPlayer* attribute), 13  
on\_base\_plus\_slugging\_percentage (*sportsreference.mlb.roster.Player* attribute), 18  
on\_base\_plus\_slugging\_percentage (*sportsreference.mlb.teams.Team* attribute), 25  
on\_base\_plus\_slugging\_percentage\_plus (*sportsreference.mlb.roster.Player* attribute), 18  
on\_base\_plus\_slugging\_percentage\_plus (*sportsreference.mlb.teams.Team* attribute), 25  
on\_court\_plus\_minus (*sportsreference.nba.roster.Player* attribute), 41  
on\_ice\_shot\_attempts\_against (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119  
on\_ice\_shot\_attempts\_for (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119  
opp\_assist\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_assists (*sportsreference.nba.teams.Team* attribute), 47  
opp\_assists (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_block\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_blocks (*sportsreference.nba.teams.Team* attribute), 47  
opp\_blocks (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_defensive\_rebounds (*sportsreference.nba.teams.Team* attribute), 47  
opp\_defensive\_rebounds (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_effective\_field\_goal\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_field\_goal\_attempts (*sportsreference.nba.teams.Team* attribute), 47  
opp\_field\_goal\_attempts (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_field\_goal\_percentage (*sportsreference.nba.teams.Team* attribute), 47  
opp\_field\_goal\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_field\_goals (*sportsreference.nba.teams.Team* attribute), 47  
opp\_field\_goals (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_free\_throw\_attempt\_rate (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_free\_throw\_attempts (*sportsreference.nba.teams.Team* attribute), 47  
opp\_free\_throw\_attempts (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_free\_throw\_percentage (*sportsreference.nba.teams.Team* attribute), 47  
opp\_free\_throw\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_free\_throws (*sportsreference.nba.teams.Team* attribute), 47  
opp\_free\_throws (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_free\_throws\_per\_field\_goal\_attempt (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_offensive\_rating (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_offensive\_rebound\_percentage (*sportsreference.ncaab.teams.Team* attribute), 69  
opp\_offensive\_rebounds (*sportsreference.nba.teams.Team* attribute), 47  
opp\_offensive\_rebounds (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_penalties\_in\_minutes (*sportsreference.nhl.schedule.Game* attribute), 129  
opp\_personal\_fouls (*sportsreference.nba.teams.Team* attribute), 47  
opp\_personal\_fouls (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_points (*sportsreference.nba.teams.Team* attribute), 47  
opp\_points (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_power\_play\_goals (*sportsreference.nhl.schedule.Game* attribute), 129  
opp\_power\_play\_opportunities (*sportsreference.nhl.schedule.Game* attribute), 129  
opp\_short\_handed\_goals (*sportsreference.nhl.schedule.Game* attribute), 129  
opp\_shots\_on\_goal (*sportsreference.nhl.schedule.Game* attribute), 129  
opp\_steal\_percentage (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_steals (*sportsreference.nba.teams.Team* attribute), 47  
opp\_steals (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_three\_point\_attempt\_rate (*sportsreference.ncaab.teams.Team* attribute), 70  
opp\_three\_point\_field\_goal\_attempts

*(sportsreference.nba.teams.Team attribute)*, 47  
 opp\_three\_point\_field\_goal\_attempts  
*(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_three\_point\_field\_goal\_percentage  
*(sportsreference.nba.teams.Team attribute)*, 47  
 opp\_three\_point\_field\_goal\_percentage  
*(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_three\_point\_field\_goals *(sportsreference.nba.teams.Team attribute)*, 47  
 opp\_three\_point\_field\_goals *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_total\_rebound\_percentage *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_total\_rebounds *(sportsreference.nba.teams.Team attribute)*, 47  
 opp\_total\_rebounds *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_true\_shooting\_percentage *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_turnover\_percentage *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_turnovers *(sportsreference.nba.teams.Team attribute)*, 47  
 opp\_turnovers *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_two\_point\_field\_goal\_attempts *(sportsreference.nba.teams.Team attribute)*, 48  
 opp\_two\_point\_field\_goal\_attempts *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_two\_point\_field\_goal\_percentage  
*(sportsreference.nba.teams.Team attribute)*, 48  
 opp\_two\_point\_field\_goal\_percentage  
*(sportsreference.ncaab.teams.Team attribute)*, 70  
 opp\_two\_point\_field\_goals *(sportsreference.nba.teams.Team attribute)*, 48  
 opp\_two\_point\_field\_goals *(sportsreference.ncaab.teams.Team attribute)*, 70  
 opponent\_abbr *(sportsreference.mlb.schedule.Game attribute)*, 21  
 opponent\_abbr *(sportsreference.nba.schedule.Game attribute)*, 44  
 opponent\_abbr *(sportsreference.ncaab.schedule.Game attribute)*, 65  
 opponent\_abbr *(sportsreference.ncaaf.schedule.Game attribute)*, 88  
 opponent\_abbr *(sportsreference.nfl.schedule.Game attribute)*, 110  
 opponent\_abbr *(sportsreference.nhl.schedule.Game attribute)*, 129  
 opponent\_conference *(sportsreference.ncaab.schedule.Game attribute)*, 65  
 opponent\_conference *(sportsreference.ncaaf.schedule.Game attribute)*, 88  
 opponent\_name *(sportsreference.nba.schedule.Game attribute)*, 44  
 opponent\_name *(sportsreference.ncaab.schedule.Game attribute)*, 65  
 opponent\_name *(sportsreference.ncaaf.schedule.Game attribute)*, 88  
 opponent\_name *(sportsreference.nfl.schedule.Game attribute)*, 110  
 opponent\_name *(sportsreference.nhl.schedule.Game attribute)*, 129  
 opponent\_rank *(sportsreference.ncaab.schedule.Game attribute)*, 65  
 opponent\_rank *(sportsreference.ncaaf.schedule.Game attribute)*, 88  
 opponents\_first\_downs *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_first\_downs\_from\_penalties  
*(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_fumbles\_lost *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_interceptions *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_attempts *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_completion\_percentage  
*(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_completions *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_first\_downs *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_touchdowns *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_pass\_yards *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_penalties *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_plays *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_rush\_attempts *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_rush\_first\_downs *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_rush\_touchdowns *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_rush\_yards *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_rush\_yards\_per\_attempt *(sportsreference.ncaaf.teams.Team attribute)*, 91  
 opponents\_turnovers *(sportsreference.ncaaf.teams.Team attribute)*, 91

opponents\_yards (*sportsreference.ncaaf.teams.Team* attribute), 92

opponents\_yards\_from\_penalties (*sportsreference.ncaaf.teams.Team* attribute), 92

opponents\_yards\_per\_play (*sportsreference.ncaaf.teams.Team* attribute), 92

opposing\_runners\_left\_on\_base (*sportsreference.mlb.teams.Team* attribute), 26

other\_touchdowns (*sportsreference.ncaaf.roster.Player* attribute), 85

other\_turnovers (*sportsreference.nba.roster.Player* attribute), 41

overtime (*sportsreference.nfl.schedule.Game* attribute), 110

overtime (*sportsreference.nhl.schedule.Game* attribute), 129

overtime\_losses (*sportsreference.nhl.teams.Team* attribute), 132

overtimes (*sportsreference.ncaab.schedule.Game* attribute), 65

## P

pace (*sportsreference.nba.boxscore.Boxscore* attribute), 33

pace (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54

pace (*sportsreference.ncaab.teams.Team* attribute), 70

pass\_attempts (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80

pass\_attempts (*sportsreference.ncaaf.roster.Player* attribute), 85

pass\_attempts (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_attempts (*sportsreference.nfl.schedule.Game* attribute), 110

pass\_attempts (*sportsreference.nfl.teams.Team* attribute), 114

pass\_completion\_percentage (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_completion\_rate (*sportsreference.nfl.schedule.Game* attribute), 111

pass\_completions (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_completions (*sportsreference.nfl.schedule.Game* attribute), 111

pass\_completions (*sportsreference.nfl.teams.Team* attribute), 114

pass\_first\_downs (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_first\_downs (*sportsreference.nfl.teams.Team* attribute), 114

pass\_net\_yards\_per\_attempt (*sportsreference.nfl.teams.Team* attribute), 114

pass\_touchdowns (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_touchdowns (*sportsreference.nfl.schedule.Game* attribute), 111

pass\_touchdowns (*sportsreference.nfl.teams.Team* attribute), 114

pass\_yards (*sportsreference.ncaaf.teams.Team* attribute), 92

pass\_yards (*sportsreference.nfl.schedule.Game* attribute), 111

pass\_yards (*sportsreference.nfl.teams.Team* attribute), 114

pass\_yards\_per\_attempt (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76

pass\_yards\_per\_attempt (*sportsreference.nfl.schedule.Game* attribute), 111

passer\_rating\_index (*sportsreference.nfl.roster.Player* attribute), 106

passes\_defended (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80

passes\_defended (*sportsreference.ncaaf.roster.Player* attribute), 85

passes\_defended (*sportsreference.nfl.player.AbstractPlayer* attribute), 101

passes\_defended (*sportsreference.nfl.roster.Player* attribute), 106

passing\_completion (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80

passing\_completion (*sportsreference.ncaaf.roster.Player* attribute), 85

passing\_completion (*sportsreference.nfl.roster.Player* attribute), 106

passing\_touchdown\_percentage (*sportsreference.nfl.roster.Player* attribute), 106

passing\_touchdowns (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80

passing\_touchdowns (*sportsreference.ncaaf.roster.Player* attribute), 85

passing\_touchdowns (*sportsreference.nfl.player.AbstractPlayer* attribute), 101

passing\_touchdowns (*sportsreference.nfl.roster.Player* attribute), 106

passing\_turnovers (*sportsreference.nba.roster.Player* attribute), 41

passing\_yards (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80

passing\_yards (*sportsreference.ncaaf.roster.Player*



attribute), 85  
 passing\_yards (sportsreference.nfl.player.AbstractPlayer attribute), 101  
 passing\_yards (sportsreference.nfl.roster.Player attribute), 106  
 passing\_yards\_per\_attempt (sportsreference.ncaaf.player.AbstractPlayer attribute), 80  
 passing\_yards\_per\_attempt (sportsreference.nfl.roster.Player attribute), 106  
 pdo (sportsreference.nhl.roster.Player attribute), 125  
 pdo (sportsreference.nhl.schedule.Game attribute), 129  
 pdo\_at\_even\_strength (sportsreference.nhl.teams.Team attribute), 132  
 penalties (sportsreference.ncaaf.teams.Team attribute), 92  
 penalties (sportsreference.nfl.teams.Team attribute), 114  
 penalties\_in\_minutes (sportsreference.nhl.player.AbstractPlayer attribute), 122  
 penalties\_in\_minutes (sportsreference.nhl.schedule.Game attribute), 130  
 penalty\_killing\_percentage (sportsreference.nhl.teams.Team attribute), 132  
 percent\_drives\_with\_points (sportsreference.nfl.teams.Team attribute), 114  
 percent\_drives\_with\_turnovers (sportsreference.nfl.teams.Team attribute), 114  
 percentage\_field\_goals\_as\_dunks (sportsreference.nba.roster.Player attribute), 41  
 percentage\_of\_three\_pointers\_from\_corner (sportsreference.nba.roster.Player attribute), 41  
 percentage\_shots\_three\_pointers (sportsreference.nba.roster.Player attribute), 41  
 percentage\_shots\_two\_pointers (sportsreference.nba.roster.Player attribute), 41  
 percentage\_sixteen\_foot\_plus\_two\_pointers (sportsreference.nba.roster.Player attribute), 41  
 percentage\_ten\_to\_sixteen\_footers (sportsreference.nba.roster.Player attribute), 41  
 percentage\_three\_to\_ten\_footers (sportsreference.nba.roster.Player attribute), 41  
 percentage\_zero\_to\_three\_footers (sportsreference.nba.roster.Player attribute), 41  
 personal\_fouls (sportsreference.nba.player.AbstractPlayer attribute), 37  
 personal\_fouls (sportsreference.nba.teams.Team attribute), 48  
 personal\_fouls (sportsreference.ncaab.player.AbstractPlayer attribute), 59  
 personal\_fouls (sportsreference.ncaab.teams.Team attribute), 70  
 pitches\_thrown (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 10  
 plate\_appearances (sportsreference.mlb.player.AbstractPlayer attribute), 13  
 plate\_appearances (sportsreference.mlb.roster.Player attribute), 18  
 plate\_appearances (sportsreference.mlb.teams.Team attribute), 26  
 Player (class in sportsreference.mlb.roster), 14  
 Player (class in sportsreference.nba.roster), 39  
 Player (class in sportsreference.ncaab.roster), 62  
 Player (class in sportsreference.ncaaf.roster), 84  
 Player (class in sportsreference.nfl.roster), 103  
 Player (class in sportsreference.nhl.roster), 123  
 player\_efficiency\_rating (sportsreference.nba.roster.Player attribute), 41  
 player\_efficiency\_rating (sportsreference.ncaab.roster.Player attribute), 63  
 player\_id (sportsreference.mlb.player.AbstractPlayer attribute), 13  
 player\_id (sportsreference.nba.player.AbstractPlayer attribute), 37  
 player\_id (sportsreference.ncaab.player.AbstractPlayer attribute), 59  
 player\_id (sportsreference.ncaaf.player.AbstractPlayer attribute), 80  
 player\_id (sportsreference.nfl.player.AbstractPlayer attribute), 101  
 player\_id (sportsreference.nhl.player.AbstractPlayer attribute), 122  
 players (sportsreference.mlb.roster.Roster attribute), 20  
 players (sportsreference.nba.roster.Roster attribute), 43  
 players (sportsreference.ncaab.roster.Roster attribute), 64  
 players (sportsreference.ncaaf.roster.Roster attribute), 87  
 players (sportsreference.nfl.roster.Roster attribute), 109  
 players (sportsreference.nhl.roster.Roster attribute), 127  
 playoff\_round (sportsreference.nhl.boxscore.Boxscore attribute), 118  
 playoffs (sportsreference.nba.schedule.Game attribute), 44

plays (*sportsreference.ncaaf.teams.Team* attribute), 92  
plays (*sportsreference.nfl.teams.Team* attribute), 114  
plays\_from\_scrimmage (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80  
plays\_from\_scrimmage (*sportsreference.ncaaf.roster.Player* attribute), 85  
plus\_minus (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
point\_guard\_percentage (*sportsreference.nba.roster.Player* attribute), 41  
point\_shares (*sportsreference.nhl.roster.Player* attribute), 125  
points (*sportsreference.nba.player.AbstractPlayer* attribute), 37  
points (*sportsreference.nba.teams.Team* attribute), 48  
points (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59  
points (*sportsreference.ncaab.teams.Team* attribute), 71  
points (*sportsreference.ncaaf.roster.Player* attribute), 85  
points (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
points (*sportsreference.nhl.teams.Team* attribute), 132  
points\_against (*sportsreference.ncaab.schedule.Game* attribute), 65  
points\_against (*sportsreference.ncaaf.schedule.Game* attribute), 88  
points\_against (*sportsreference.nfl.teams.Team* attribute), 114  
points\_against\_per\_game (*sportsreference.ncaaf.teams.Team* attribute), 92  
points\_allowed (*sportsreference.nba.schedule.Game* attribute), 44  
points\_allowed (*sportsreference.nfl.schedule.Game* attribute), 111  
points\_contributed\_by\_offense (*sportsreference.nfl.teams.Team* attribute), 114  
points\_difference (*sportsreference.nfl.teams.Team* attribute), 114  
points\_for (*sportsreference.ncaab.schedule.Game* attribute), 65  
points\_for (*sportsreference.ncaaf.schedule.Game* attribute), 88  
points\_for (*sportsreference.nfl.teams.Team* attribute), 114  
points\_generated\_by\_assists (*sportsreference.nba.roster.Player* attribute), 41  
points\_kicking (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
points\_per\_game (*sportsreference.ncaaf.teams.Team* attribute), 92  
points\_percentage (*sportsreference.nhl.teams.Team* attribute), 132  
points\_produced (*sportsreference.ncaab.roster.Player* attribute), 63  
points\_scored (*sportsreference.nba.schedule.Game* attribute), 44  
points\_scored (*sportsreference.nfl.schedule.Game* attribute), 111  
position (*sportsreference.mlb.roster.Player* attribute), 18  
position (*sportsreference.nba.roster.Player* attribute), 41  
position (*sportsreference.ncaab.roster.Player* attribute), 63  
position (*sportsreference.ncaaf.roster.Player* attribute), 85  
position (*sportsreference.nfl.roster.Player* attribute), 106  
power\_forward\_percentage (*sportsreference.nba.roster.Player* attribute), 41  
power\_play\_assists (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
power\_play\_goals (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
power\_play\_goals (*sportsreference.nhl.schedule.Game* attribute), 130  
power\_play\_goals (*sportsreference.nhl.teams.Team* attribute), 132  
power\_play\_goals\_against (*sportsreference.nhl.teams.Team* attribute), 132  
power\_play\_goals\_against\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 125  
power\_play\_goals\_allowed (*sportsreference.nhl.roster.Player* attribute), 125  
power\_play\_goals\_for\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 126  
power\_play\_opportunities (*sportsreference.nhl.schedule.Game* attribute), 130  
power\_play\_opportunities (*sportsreference.nhl.teams.Team* attribute), 132  
power\_play\_opportunities\_against (*sportsreference.nhl.teams.Team* attribute), 132  
power\_play\_percentage (*sportsreference.nhl.teams.Team* attribute), 132  
power\_play\_save\_percentage (*sportsreference.nhl.roster.Player* attribute), 126  
power\_play\_shots\_faced (*sportsreference.nhl.roster.Player* attribute), 126  
punt\_return\_touchdown (*sportsreference.nfl.player.AbstractPlayer* attribute), 101

punt\_return\_touchdown (*sportsreference.nfl.roster.Player* attribute), 106  
 punt\_return\_touchdowns (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 80  
 punt\_return\_touchdowns (*sportsreference.ncaaf.roster.Player* attribute), 85  
 punt\_return\_yards (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
 punt\_return\_yards (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 punt\_return\_yards (*sportsreference.nfl.roster.Player* attribute), 106  
 punt\_returns (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
 punt\_returns (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 punt\_returns (*sportsreference.nfl.roster.Player* attribute), 106  
 punt\_yards (*sportsreference.nfl.schedule.Game* attribute), 111  
 punting\_yards (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
 punting\_yards\_per\_attempt (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
 punts (*sportsreference.ncaaf.boxscore.BoxscorePlayer* attribute), 76  
 punts (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 punts (*sportsreference.nfl.roster.Player* attribute), 106  
 punts (*sportsreference.nfl.schedule.Game* attribute), 111  
 putouts (*sportsreference.mlb.player.AbstractPlayer* attribute), 13  
 putouts (*sportsreference.mlb.roster.Player* attribute), 18  
 pythagorean\_win\_loss (*sportsreference.mlb.teams.Team* attribute), 26

## Q

qb\_record (*sportsreference.nfl.roster.Player* attribute), 106  
 quality\_start\_percentage (*sportsreference.nhl.roster.Player* attribute), 126  
 quality\_starts (*sportsreference.nhl.roster.Player* attribute), 126  
 quarterback\_hits (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute),

98  
 quarterback\_rating (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81  
 quarterback\_rating (*sportsreference.ncaaf.roster.Player* attribute), 85  
 quarterback\_rating (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 quarterback\_rating (*sportsreference.nfl.roster.Player* attribute), 107  
 quarterback\_rating (*sportsreference.nfl.schedule.Game* attribute), 111

## R

range\_factor\_per\_game (*sportsreference.mlb.roster.Player* attribute), 18  
 range\_factor\_per\_nine\_innings (*sportsreference.mlb.roster.Player* attribute), 18  
 rank (*sportsreference.mlb.schedule.Game* attribute), 21  
 rank (*sportsreference.mlb.teams.Team* attribute), 26  
 rank (*sportsreference.nba.teams.Team* attribute), 48  
 rank (*sportsreference.ncaaf.schedule.Game* attribute), 88  
 rank (*sportsreference.nfl.teams.Team* attribute), 114  
 rank (*sportsreference.nhl.teams.Team* attribute), 132  
 Rankings (class in *sportsreference.ncaab.rankings*), 60  
 Rankings (class in *sportsreference.ncaaf.rankings*), 82  
 really\_bad\_starts (*sportsreference.nhl.roster.Player* attribute), 126  
 receiving\_touchdowns (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81  
 receiving\_touchdowns (*sportsreference.ncaaf.roster.Player* attribute), 85  
 receiving\_touchdowns (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 receiving\_touchdowns (*sportsreference.nfl.roster.Player* attribute), 107  
 receiving\_yards (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81  
 receiving\_yards (*sportsreference.ncaaf.roster.Player* attribute), 86  
 receiving\_yards (*sportsreference.nfl.player.AbstractPlayer* attribute), 101  
 receiving\_yards (*sportsreference.nfl.roster.Player* attribute), 107  
 receiving\_yards\_per\_game (*sportsreference.nfl.roster.Player* attribute), 107  
 receiving\_yards\_per\_reception (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

receiving\_yards\_per\_reception (*sportsreference.ncaaf.roster.Player* attribute), 86

receiving\_yards\_per\_reception (*sportsreference.nfl.roster.Player* attribute), 107

receptions (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

receptions (*sportsreference.ncaaf.roster.Player* attribute), 86

receptions (*sportsreference.nfl.player.AbstractPlayer* attribute), 101

receptions (*sportsreference.nfl.roster.Player* attribute), 107

receptions\_per\_game (*sportsreference.nfl.roster.Player* attribute), 107

record (*sportsreference.mlb.schedule.Game* attribute), 21

record\_vs\_left\_handed\_pitchers (*sportsreference.mlb.teams.Team* attribute), 26

record\_vs\_right\_handed\_pitchers (*sportsreference.mlb.teams.Team* attribute), 26

record\_vs\_teams\_over\_500 (*sportsreference.mlb.teams.Team* attribute), 26

record\_vs\_teams\_under\_500 (*sportsreference.mlb.teams.Team* attribute), 26

relative\_corsi\_for\_percentage (*sportsreference.nhl.player.AbstractPlayer* attribute), 122

relative\_fenwick\_for\_percentage (*sportsreference.nhl.roster.Player* attribute), 126

result (*sportsreference.mlb.schedule.Game* attribute), 21

result (*sportsreference.nba.schedule.Game* attribute), 44

result (*sportsreference.ncaab.schedule.Game* attribute), 65

result (*sportsreference.ncaaf.schedule.Game* attribute), 88

result (*sportsreference.nfl.schedule.Game* attribute), 111

result (*sportsreference.nhl.schedule.Game* attribute), 130

Roster (class in *sportsreference.mlb.roster*), 19

Roster (class in *sportsreference.nba.roster*), 43

Roster (class in *sportsreference.ncaab.roster*), 64

Roster (class in *sportsreference.ncaaf.roster*), 87

Roster (class in *sportsreference.nfl.roster*), 109

Roster (class in *sportsreference.nhl.roster*), 127

roster (*sportsreference.mlb.teams.Team* attribute), 26

roster (*sportsreference.nba.teams.Team* attribute), 48

roster (*sportsreference.ncaab.teams.Team* attribute), 71

roster (*sportsreference.ncaaf.teams.Team* attribute), 92

roster (*sportsreference.nfl.teams.Team* attribute), 114

roster (*sportsreference.nhl.teams.Team* attribute), 132

run\_difference (*sportsreference.mlb.teams.Team* attribute), 26

runners\_left\_on\_base (*sportsreference.mlb.teams.Team* attribute), 26

runs (*sportsreference.mlb.player.AbstractPlayer* attribute), 13

runs (*sportsreference.mlb.roster.Player* attribute), 18

runs (*sportsreference.mlb.teams.Team* attribute), 26

runs\_against (*sportsreference.mlb.teams.Team* attribute), 26

runs\_allowed (*sportsreference.mlb.player.AbstractPlayer* attribute), 13

runs\_allowed (*sportsreference.mlb.roster.Player* attribute), 18

runs\_allowed (*sportsreference.mlb.schedule.Game* attribute), 21

runs\_allowed\_per\_game (*sportsreference.mlb.teams.Team* attribute), 26

runs\_batted\_in (*sportsreference.mlb.player.AbstractPlayer* attribute), 13

runs\_batted\_in (*sportsreference.mlb.roster.Player* attribute), 18

runs\_batted\_in (*sportsreference.mlb.teams.Team* attribute), 26

runs\_scored (*sportsreference.mlb.schedule.Game* attribute), 21

rush\_attempts (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

rush\_attempts (*sportsreference.ncaaf.roster.Player* attribute), 86

rush\_attempts (*sportsreference.ncaaf.teams.Team* attribute), 92

rush\_attempts (*sportsreference.nfl.player.AbstractPlayer* attribute), 101

rush\_attempts (*sportsreference.nfl.roster.Player* attribute), 107

rush\_attempts (*sportsreference.nfl.schedule.Game* attribute), 111

rush\_attempts (*sportsreference.nfl.teams.Team* attribute), 114

rush\_attempts\_per\_game (*sportsreference.nfl.roster.Player* attribute), 107

rush\_first\_downs (*sportsreference.ncaaf.teams.Team* attribute), 92

rush\_first\_downs (*sportsreference.nfl.teams.Team* attribute), 114

rush\_touchdowns (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

rush\_touchdowns (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 rush\_touchdowns (*sportsreference.ncaa.f.teams.Team attribute*), 92  
 rush\_touchdowns (*sportsreference.nfl.player.AbstractPlayer attribute*), 101  
 rush\_touchdowns (*sportsreference.nfl.roster.Player attribute*), 107  
 rush\_touchdowns (*sportsreference.nfl.schedule.Game attribute*), 111  
 rush\_touchdowns (*sportsreference.nfl.teams.Team attribute*), 114  
 rush\_yards (*sportsreference.ncaa.f.player.AbstractPlayer attribute*), 81  
 rush\_yards (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 rush\_yards (*sportsreference.ncaa.f.teams.Team attribute*), 92  
 rush\_yards (*sportsreference.nfl.player.AbstractPlayer attribute*), 101  
 rush\_yards (*sportsreference.nfl.roster.Player attribute*), 107  
 rush\_yards (*sportsreference.nfl.schedule.Game attribute*), 111  
 rush\_yards (*sportsreference.nfl.teams.Team attribute*), 114  
 rush\_yards\_per\_attempt (*sportsreference.ncaa.f.player.AbstractPlayer attribute*), 81  
 rush\_yards\_per\_attempt (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 rush\_yards\_per\_attempt (*sportsreference.ncaa.f.teams.Team attribute*), 92  
 rush\_yards\_per\_attempt (*sportsreference.nfl.roster.Player attribute*), 107  
 rush\_yards\_per\_attempt (*sportsreference.nfl.schedule.Game attribute*), 111  
 rush\_yards\_per\_attempt (*sportsreference.nfl.teams.Team attribute*), 115  
 rush\_yards\_per\_game (*sportsreference.nfl.roster.Player attribute*), 107  
 rushing\_and\_receiving\_touchdowns (*sportsreference.ncaa.f.player.AbstractPlayer attribute*), 81  
 rushing\_and\_receiving\_touchdowns (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 rushing\_and\_receiving\_touchdowns (*sportsreference.nfl.roster.Player attribute*), 107  
 sack\_percentage\_index (*sportsreference.nfl.roster.Player attribute*), 107  
 sacks (*sportsreference.ncaa.f.player.AbstractPlayer attribute*), 81  
 sacks (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 sacks (*sportsreference.nfl.player.AbstractPlayer attribute*), 101  
 sacks (*sportsreference.nfl.roster.Player attribute*), 107  
 sacrifice\_flies (*sportsreference.mlb.roster.Player attribute*), 18  
 sacrifice\_flies (*sportsreference.mlb.teams.Team attribute*), 26  
 sacrifice\_hits (*sportsreference.mlb.roster.Player attribute*), 18  
 sacrifice\_hits (*sportsreference.mlb.teams.Team attribute*), 26  
 safeties (*sportsreference.ncaa.f.roster.Player attribute*), 86  
 safeties (*sportsreference.nfl.roster.Player attribute*), 107  
 salary (*sportsreference.nba.roster.Player attribute*), 42  
 save (*sportsreference.mlb.schedule.Game attribute*), 21  
 save\_percentage (*sportsreference.nhl.player.AbstractPlayer attribute*), 122  
 save\_percentage (*sportsreference.nhl.teams.Team attribute*), 132  
 save\_percentage\_on\_ice (*sportsreference.nhl.roster.Player attribute*), 126  
 saves (*sportsreference.mlb.roster.Player attribute*), 18  
 saves (*sportsreference.mlb.teams.Team attribute*), 26  
 saves (*sportsreference.nhl.player.AbstractPlayer attribute*), 122  
 Schedule (*class in sportsreference.mlb.schedule*), 21  
 Schedule (*class in sportsreference.nba.schedule*), 45  
 Schedule (*class in sportsreference.ncaab.schedule*), 66  
 Schedule (*class in sportsreference.ncaa.f.schedule*), 89  
 Schedule (*class in sportsreference.nfl.schedule*), 112  
 Schedule (*class in sportsreference.nhl.schedule*), 130  
 schedule (*sportsreference.mlb.teams.Team attribute*), 26  
 schedule (*sportsreference.nba.teams.Team attribute*), 48  
 schedule (*sportsreference.ncaab.teams.Team attribute*), 71  
 schedule (*sportsreference.ncaa.f.teams.Team attribute*), 92  
 schedule (*sportsreference.nfl.teams.Team attribute*), 115  
 schedule (*sportsreference.nhl.teams.Team attribute*), 132  
 season (*sportsreference.mlb.roster.Player attribute*), 18  
 season (*sportsreference.nba.roster.Player attribute*), 42  
 season (*sportsreference.ncaab.roster.Player attribute*), 66

## S



63  
season (*sportsreference.ncaaf.roster.Player* attribute), 86  
season (*sportsreference.nfl.roster.Player* attribute), 107  
season (*sportsreference.nhl.roster.Player* attribute), 126  
season\_losses (*sportsreference.ncaab.schedule.Game* attribute), 65  
season\_wins (*sportsreference.ncaab.schedule.Game* attribute), 66  
shifts (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119  
shooting\_distance (*sportsreference.nba.roster.Player* attribute), 42  
shooting\_fouls (*sportsreference.nba.roster.Player* attribute), 42  
shooting\_fouls\_drawn (*sportsreference.nba.roster.Player* attribute), 42  
shooting\_guard\_percentage (*sportsreference.nba.roster.Player* attribute), 42  
shooting\_percentage (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
shooting\_percentage (*sportsreference.nhl.teams.Team* attribute), 132  
shooting\_percentage\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 126  
shootout\_attempts (*sportsreference.nhl.roster.Player* attribute), 126  
shootout\_goals (*sportsreference.nhl.roster.Player* attribute), 126  
shootout\_misses (*sportsreference.nhl.roster.Player* attribute), 126  
shootout\_percentage (*sportsreference.nhl.roster.Player* attribute), 126  
short\_handed\_assists (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
short\_handed\_goals (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
short\_handed\_goals (*sportsreference.nhl.schedule.Game* attribute), 130  
short\_handed\_goals (*sportsreference.nhl.teams.Team* attribute), 132  
short\_handed\_goals\_against (*sportsreference.nhl.teams.Team* attribute), 132  
short\_handed\_goals\_allowed (*sportsreference.nhl.roster.Player* attribute), 126  
short\_handed\_save\_percentage (*sportsreference.nhl.roster.Player* attribute), 126  
short\_handed\_shots\_faced (*sportsreference.nhl.roster.Player* attribute), 126  
shots\_against (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
shots\_against (*sportsreference.nhl.teams.Team* attribute), 132  
shots\_blocked (*sportsreference.nba.roster.Player* attribute), 42  
shots\_on\_goal (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
shots\_on\_goal (*sportsreference.nhl.schedule.Game* attribute), 130  
shots\_on\_goal (*sportsreference.nhl.teams.Team* attribute), 132  
shutouts (*sportsreference.mlb.roster.Player* attribute), 18  
shutouts (*sportsreference.mlb.teams.Team* attribute), 27  
shutouts (*sportsreference.nhl.player.AbstractPlayer* attribute), 122  
simple\_rating\_system (*sportsreference.mlb.teams.Team* attribute), 27  
simple\_rating\_system (*sportsreference.ncaab.teams.Team* attribute), 71  
simple\_rating\_system (*sportsreference.ncaaf.teams.Team* attribute), 92  
simple\_rating\_system (*sportsreference.nfl.teams.Team* attribute), 115  
simple\_rating\_system (*sportsreference.nhl.teams.Team* attribute), 133  
single\_run\_losses (*sportsreference.mlb.teams.Team* attribute), 27  
single\_run\_record (*sportsreference.mlb.teams.Team* attribute), 27  
single\_run\_wins (*sportsreference.mlb.teams.Team* attribute), 27  
slugging\_percentage (*sportsreference.mlb.player.AbstractPlayer* attribute), 13  
slugging\_percentage (*sportsreference.mlb.roster.Player* attribute), 18  
slugging\_percentage (*sportsreference.mlb.teams.Team* attribute), 27  
small\_forward\_percentage (*sportsreference.nba.roster.Player* attribute), 42  
solo\_tackles (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81  
solo\_tackles (*sportsreference.ncaaf.roster.Player* attribute), 86  
solo\_tackles (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute), 98  
sportsreference.mlb.boxscore (module), 5  
sportsreference.mlb.player (module), 12

- sportsreference.mlb.roster (module), 14
- sportsreference.mlb.schedule (module), 20
- sportsreference.mlb.teams (module), 22
- sportsreference.nba.boxscore (module), 29
- sportsreference.nba.player (module), 36
- sportsreference.nba.roster (module), 39
- sportsreference.nba.schedule (module), 43
- sportsreference.nba.teams (module), 46
- sportsreference.ncaab.boxscore (module), 49
- sportsreference.ncaab.conferences (module), 57
- sportsreference.ncaab.player (module), 58
- sportsreference.ncaab.rankings (module), 60
- sportsreference.ncaab.roster (module), 62
- sportsreference.ncaab.schedule (module), 64
- sportsreference.ncaab.teams (module), 67
- sportsreference.ncaaf.boxscore (module), 73
- sportsreference.ncaaf.conferences (module), 78
- sportsreference.ncaaf.player (module), 79
- sportsreference.ncaaf.rankings (module), 82
- sportsreference.ncaaf.roster (module), 84
- sportsreference.ncaaf.schedule (module), 87
- sportsreference.ncaaf.teams (module), 90
- sportsreference.nfl.boxscore (module), 94
- sportsreference.nfl.player (module), 99
- sportsreference.nfl.roster (module), 103
- sportsreference.nfl.schedule (module), 109
- sportsreference.nfl.teams (module), 113
- sportsreference.nhl.boxscore (module), 116
- sportsreference.nhl.player (module), 121
- sportsreference.nhl.roster (module), 123
- sportsreference.nhl.schedule (module), 128
- sportsreference.nhl.teams (module), 131
- stadium (sportsreference.ncaaf.boxscore.Boxscore attribute), 75
- stadium (sportsreference.nfl.boxscore.Boxscore attribute), 97
- steal\_percentage (sportsreference.nba.player.AbstractPlayer attribute), 37
- steal\_percentage (sportsreference.ncaab.player.AbstractPlayer attribute), 59
- steal\_percentage (sportsreference.ncaab.teams.Team attribute), 71
- steals (sportsreference.nba.player.AbstractPlayer attribute), 37
- steals (sportsreference.nba.teams.Team attribute), 48
- steals (sportsreference.ncaab.player.AbstractPlayer attribute), 59
- steals (sportsreference.ncaab.teams.Team attribute), 71
- stolen\_bases (sportsreference.mlb.roster.Player attribute), 18
- stolen\_bases (sportsreference.mlb.teams.Team attribute), 27
- streak (sportsreference.mlb.schedule.Game attribute), 21
- streak (sportsreference.mlb.teams.Team attribute), 27
- streak (sportsreference.nba.schedule.Game attribute), 44
- streak (sportsreference.ncaab.schedule.Game attribute), 66
- streak (sportsreference.ncaaf.schedule.Game attribute), 89
- strength\_of\_schedule (sportsreference.mlb.teams.Team attribute), 27
- strength\_of\_schedule (sportsreference.ncaab.teams.Team attribute), 71
- strength\_of\_schedule (sportsreference.ncaaf.teams.Team attribute), 93
- strength\_of\_schedule (sportsreference.nfl.teams.Team attribute), 115
- strength\_of\_schedule (sportsreference.nhl.teams.Team attribute), 133
- strikeouts (sportsreference.mlb.player.AbstractPlayer attribute), 13
- strikeouts (sportsreference.mlb.roster.Player attribute), 18
- strikeouts (sportsreference.mlb.teams.Team attribute), 27
- strikeouts\_per\_base\_on\_balls (sportsreference.mlb.teams.Team attribute), 27
- strikeouts\_per\_nine\_innings (sportsreference.mlb.teams.Team attribute), 27
- strikeouts\_thrown\_per\_walk (sportsreference.mlb.roster.Player attribute), 18
- strikes (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 11
- strikes\_contact (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 11
- strikes\_looking (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 11
- strikes\_swinging (sportsreference.mlb.boxscore.BoxscorePlayer attribute), 11
- strikes\_thrown (sportsreference.mlb.boxscore.BoxscorePlayer attribute),

11

**T**

- tackles (*sportsreference.nfl.roster.Player* attribute), 107
- tackles\_for\_loss (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81
- tackles\_for\_loss (*sportsreference.ncaaf.roster.Player* attribute), 86
- tackles\_for\_loss (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute), 98
- take\_fouls (*sportsreference.nba.roster.Player* attribute), 42
- takeaways (*sportsreference.nhl.roster.Player* attribute), 126
- Team (class in *sportsreference.mlb.teams*), 22
- Team (class in *sportsreference.nba.teams*), 46
- Team (class in *sportsreference.ncaab.teams*), 67
- Team (class in *sportsreference.ncaaf.teams*), 90
- Team (class in *sportsreference.nfl.teams*), 113
- Team (class in *sportsreference.nhl.teams*), 131
- team\_abbreviation (*sportsreference.mlb.roster.Player* attribute), 19
- team\_abbreviation (*sportsreference.nba.roster.Player* attribute), 42
- team\_abbreviation (*sportsreference.ncaab.roster.Player* attribute), 63
- team\_abbreviation (*sportsreference.ncaaf.roster.Player* attribute), 86
- team\_abbreviation (*sportsreference.nfl.roster.Player* attribute), 108
- team\_abbreviation (*sportsreference.nhl.roster.Player* attribute), 126
- team\_conference (*sportsreference.ncaab.conferences.Conferences* attribute), 57
- team\_conference (*sportsreference.ncaaf.conferences.Conferences* attribute), 79
- Teams (class in *sportsreference.mlb.teams*), 28
- Teams (class in *sportsreference.nba.teams*), 48
- Teams (class in *sportsreference.ncaab.teams*), 72
- Teams (class in *sportsreference.ncaaf.teams*), 93
- Teams (class in *sportsreference.nfl.teams*), 115
- Teams (class in *sportsreference.nhl.teams*), 133
- teams (*sportsreference.ncaab.conferences.Conference* attribute), 57
- teams (*sportsreference.ncaaf.conferences.Conference* attribute), 78
- third\_down\_attempts (*sportsreference.nfl.schedule.Game* attribute), 111
- third\_down\_conversions (*sportsreference.nfl.schedule.Game* attribute), 111
- thirty\_to\_thirty\_nine\_yard\_field\_goal\_attempts (*sportsreference.nfl.roster.Player* attribute), 108
- thirty\_to\_thirty\_nine\_yard\_field\_goals\_made (*sportsreference.nfl.roster.Player* attribute), 108
- three\_point\_attempt\_rate (*sportsreference.nba.player.AbstractPlayer* attribute), 37
- three\_point\_attempt\_rate (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59
- three\_point\_attempt\_rate (*sportsreference.ncaab.teams.Team* attribute), 71
- three\_point\_attempts (*sportsreference.nba.player.AbstractPlayer* attribute), 37
- three\_point\_attempts (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59
- three\_point\_field\_goal\_attempts (*sportsreference.nba.teams.Team* attribute), 48
- three\_point\_field\_goal\_attempts (*sportsreference.ncaab.teams.Team* attribute), 71
- three\_point\_field\_goal\_percentage (*sportsreference.nba.teams.Team* attribute), 48
- three\_point\_field\_goal\_percentage (*sportsreference.ncaab.teams.Team* attribute), 71
- three\_point\_field\_goals (*sportsreference.nba.teams.Team* attribute), 48
- three\_point\_field\_goals (*sportsreference.ncaab.teams.Team* attribute), 71
- three\_point\_percentage (*sportsreference.nba.player.AbstractPlayer* attribute), 37
- three\_point\_percentage (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59
- three\_point\_shot\_percentage\_from\_corner (*sportsreference.nba.roster.Player* attribute), 42
- three\_pointers (*sportsreference.nba.player.AbstractPlayer* attribute), 38
- three\_pointers (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59
- three\_pointers\_assisted\_percentage (*sportsreference.nba.roster.Player* attribute), 42
- ties\_plus\_overtime\_loss (*sportsreference.nhl.roster.Player* attribute), 127



time (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
 time (*sportsreference.nba.schedule.Game* attribute), 44  
 time (*sportsreference.ncaab.schedule.Game* attribute), 66  
 time (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
 time (*sportsreference.ncaaf.schedule.Game* attribute), 89  
 time (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
 time (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
 time\_of\_day (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
 time\_of\_possession (*sportsreference.nfl.schedule.Game* attribute), 111  
 time\_on\_ice (*sportsreference.nhl.boxscore.BoxscorePlayer* attribute), 119  
 time\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 127  
 time\_on\_ice\_even\_strength (*sportsreference.nhl.roster.Player* attribute), 127  
 times\_caught\_stealing (*sportsreference.mlb.roster.Player* attribute), 19  
 times\_caught\_stealing (*sportsreference.mlb.teams.Team* attribute), 27  
 times\_hit\_by\_pitch (*sportsreference.mlb.roster.Player* attribute), 19  
 times\_hit\_by\_pitch (*sportsreference.mlb.teams.Team* attribute), 27  
 times\_hit\_player (*sportsreference.mlb.roster.Player* attribute), 19  
 times\_pass\_target (*sportsreference.nfl.player.AbstractPlayer* attribute), 102  
 times\_pass\_target (*sportsreference.nfl.roster.Player* attribute), 108  
 times\_sacked (*sportsreference.nfl.player.AbstractPlayer* attribute), 102  
 times\_sacked (*sportsreference.nfl.roster.Player* attribute), 108  
 times\_sacked (*sportsreference.nfl.schedule.Game* attribute), 111  
 times\_struck\_out (*sportsreference.mlb.player.AbstractPlayer* attribute), 13  
 times\_struck\_out (*sportsreference.mlb.roster.Player* attribute), 19  
 times\_struck\_out (*sportsreference.mlb.teams.Team* attribute), 27  
 total\_bases (*sportsreference.mlb.roster.Player* attribute), 19  
 total\_bases (*sportsreference.mlb.teams.Team* attribute), 27  
 total\_fielding\_runs\_above\_average (*sportsreference.mlb.roster.Player* attribute), 19  
 total\_fielding\_runs\_above\_average\_per\_innings (*sportsreference.mlb.roster.Player* attribute), 19  
 total\_goals\_against\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 127  
 total\_goals\_for\_on\_ice (*sportsreference.nhl.roster.Player* attribute), 127  
 total\_goals\_per\_game (*sportsreference.nhl.teams.Team* attribute), 133  
 total\_punt\_yards (*sportsreference.nfl.player.AbstractPlayer* attribute), 102  
 total\_punt\_yards (*sportsreference.nfl.roster.Player* attribute), 108  
 total\_rebound\_percentage (*sportsreference.nba.player.AbstractPlayer* attribute), 38  
 total\_rebound\_percentage (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59  
 total\_rebound\_percentage (*sportsreference.ncaab.teams.Team* attribute), 71  
 total\_rebounds (*sportsreference.nba.player.AbstractPlayer* attribute), 38  
 total\_rebounds (*sportsreference.nba.teams.Team* attribute), 48  
 total\_rebounds (*sportsreference.ncaab.player.AbstractPlayer* attribute), 59  
 total\_rebounds (*sportsreference.ncaab.teams.Team* attribute), 71  
 total\_runs (*sportsreference.mlb.teams.Team* attribute), 27  
 total\_shots (*sportsreference.nhl.roster.Player* attribute), 127  
 total\_tackles (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81  
 total\_tackles (*sportsreference.ncaaf.roster.Player* attribute), 86  
 total\_touchdowns (*sportsreference.ncaaf.roster.Player* attribute), 86  
 touchdown\_percentage\_index (*sportsreference.nfl.roster.Player* attribute), 108  
 touches (*sportsreference.nfl.roster.Player* attribute), 108  
 triples (*sportsreference.mlb.roster.Player* attribute), 19

`triples` (*sportsreference.mlb.teams.Team* attribute), 27

`true_shooting_percentage` (*sportsreference.nba.player.AbstractPlayer* attribute), 38

`true_shooting_percentage` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`true_shooting_percentage` (*sportsreference.ncaab.teams.Team* attribute), 71

`turnover_percentage` (*sportsreference.nba.player.AbstractPlayer* attribute), 38

`turnover_percentage` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`turnover_percentage` (*sportsreference.ncaab.teams.Team* attribute), 71

`turnovers` (*sportsreference.nba.player.AbstractPlayer* attribute), 38

`turnovers` (*sportsreference.nba.teams.Team* attribute), 48

`turnovers` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`turnovers` (*sportsreference.ncaab.teams.Team* attribute), 71

`turnovers` (*sportsreference.ncaaf.teams.Team* attribute), 93

`turnovers` (*sportsreference.nfl.teams.Team* attribute), 115

`twenty_to_twenty_nine_yard_field_goal_attempts` (*sportsreference.nfl.roster.Player* attribute), 108

`twenty_to_twenty_nine_yard_field_goals_made` (*sportsreference.nfl.roster.Player* attribute), 108

`two_point_attempts` (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 34

`two_point_attempts` (*sportsreference.nba.roster.Player* attribute), 42

`two_point_attempts` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`two_point_conversions` (*sportsreference.ncaaf.roster.Player* attribute), 86

`two_point_field_goal_attempts` (*sportsreference.nba.teams.Team* attribute), 48

`two_point_field_goal_attempts` (*sportsreference.ncaab.teams.Team* attribute), 71

`two_point_field_goal_percentage` (*sportsreference.nba.teams.Team* attribute), 48

`two_point_field_goal_percentage` (*sportsreference.ncaab.teams.Team* attribute), 71

`two_point_field_goals` (*sportsreference.nba.teams.Team* attribute), 48

`two_point_field_goals` (*sportsreference.ncaab.teams.Team* attribute), 72

`two_point_percentage` (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 34

`two_point_percentage` (*sportsreference.nba.roster.Player* attribute), 42

`two_point_percentage` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`two_pointers` (*sportsreference.nba.boxscore.BoxscorePlayer* attribute), 35

`two_pointers` (*sportsreference.nba.roster.Player* attribute), 42

`two_pointers` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

`two_pointers_assisted_percentage` (*sportsreference.nba.roster.Player* attribute), 42

`type` (*sportsreference.ncaab.schedule.Game* attribute), 66

`type` (*sportsreference.nfl.schedule.Game* attribute), 111

## U

`unknown_bat_types` (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 11

`usage_percentage` (*sportsreference.nba.player.AbstractPlayer* attribute), 38

`usage_percentage` (*sportsreference.ncaab.player.AbstractPlayer* attribute), 60

## V

`value_over_replacement_player` (*sportsreference.nba.roster.Player* attribute), 42

`venue` (*sportsreference.mlb.boxscore.Boxscore* attribute), 9

## W

`week` (*sportsreference.nfl.schedule.Game* attribute), 111

`weight` (*sportsreference.mlb.roster.Player* attribute), 19

`weight` (*sportsreference.nba.roster.Player* attribute), 43

`weight` (*sportsreference.ncaab.roster.Player* attribute), 64

`weight` (*sportsreference.ncaaf.roster.Player* attribute), 86

`weight` (*sportsreference.nfl.roster.Player* attribute), 108

`weight` (*sportsreference.nhl.roster.Player* attribute), 127

- whip (*sportsreference.mlb.roster.Player* attribute), 19  
 whip (*sportsreference.mlb.teams.Team* attribute), 27  
 wild\_pitches (*sportsreference.mlb.roster.Player* attribute), 19  
 wild\_pitches (*sportsreference.mlb.teams.Team* attribute), 27  
 win\_percentage (*sportsreference.mlb.roster.Player* attribute), 19  
 win\_percentage (*sportsreference.mlb.teams.Team* attribute), 27  
 win\_percentage (*sportsreference.ncaab.teams.Team* attribute), 72  
 win\_percentage (*sportsreference.ncaaf.teams.Team* attribute), 93  
 win\_percentage (*sportsreference.nfl.teams.Team* attribute), 115  
 win\_probability\_added (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 11  
 win\_probability\_added\_pitcher (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 11  
 win\_probability\_for\_offensive\_player (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 11  
 win\_probability\_subtracted (*sportsreference.mlb.boxscore.BoxscorePlayer* attribute), 11  
 win\_shares (*sportsreference.nba.roster.Player* attribute), 43  
 win\_shares (*sportsreference.ncaab.roster.Player* attribute), 64  
 win\_shares\_per\_40\_minutes (*sportsreference.ncaab.roster.Player* attribute), 64  
 win\_shares\_per\_48\_minutes (*sportsreference.nba.roster.Player* attribute), 43  
 winner (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
 winner (*sportsreference.mlb.schedule.Game* attribute), 21  
 winner (*sportsreference.nba.boxscore.Boxscore* attribute), 34  
 winner (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
 winner (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
 winner (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
 winner (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
 winning\_abbr (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
 winning\_abbr (*sportsreference.nba.boxscore.Boxscore* attribute), 34  
 winning\_abbr (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
 winning\_abbr (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
 winning\_abbr (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
 winning\_abbr (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
 winning\_name (*sportsreference.mlb.boxscore.Boxscore* attribute), 9  
 winning\_name (*sportsreference.nba.boxscore.Boxscore* attribute), 34  
 winning\_name (*sportsreference.ncaab.boxscore.Boxscore* attribute), 54  
 winning\_name (*sportsreference.ncaaf.boxscore.Boxscore* attribute), 75  
 winning\_name (*sportsreference.nfl.boxscore.Boxscore* attribute), 97  
 winning\_name (*sportsreference.nhl.boxscore.Boxscore* attribute), 118  
 wins (*sportsreference.mlb.roster.Player* attribute), 19  
 wins (*sportsreference.mlb.teams.Team* attribute), 28  
 wins (*sportsreference.nba.schedule.Game* attribute), 44  
 wins (*sportsreference.ncaab.teams.Team* attribute), 72  
 wins (*sportsreference.ncaaf.schedule.Game* attribute), 89  
 wins (*sportsreference.ncaaf.teams.Team* attribute), 93  
 wins (*sportsreference.nfl.teams.Team* attribute), 115  
 wins (*sportsreference.nhl.roster.Player* attribute), 127  
 wins (*sportsreference.nhl.teams.Team* attribute), 133  
 wins\_last\_ten\_games (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_last\_thirty\_games (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_last\_twenty\_games (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_vs\_left\_handed\_pitchers (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_vs\_right\_handed\_pitchers (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_vs\_teams\_over\_500 (*sportsreference.mlb.teams.Team* attribute), 28  
 wins\_vs\_teams\_under\_500 (*sportsreference.mlb.teams.Team* attribute), 28
- Y**  
 yards (*sportsreference.ncaaf.teams.Team* attribute), 93  
 yards (*sportsreference.nfl.teams.Team* attribute), 115  
 yards\_from\_penalties (*sportsreference.ncaaf.teams.Team* attribute), 93

`yards_from_penalties` (*sportsreference.nfl.teams.Team* attribute), 115

`yards_from_scrimmage` (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

`yards_from_scrimmage` (*sportsreference.ncaaf.roster.Player* attribute), 86

`yards_from_scrimmage` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_from_scrimmage_per_play` (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

`yards_from_scrimmage_per_play` (*sportsreference.ncaaf.roster.Player* attribute), 86

`yards_lost_from_sacks` (*sportsreference.nfl.boxscore.BoxscorePlayer* attribute), 98

`yards_lost_from_sacks` (*sportsreference.nfl.schedule.Game* attribute), 112

`yards_lost_to_sacks` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_attempt_index` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_completed_pass` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_game_played` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_kickoff_return` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_play` (*sportsreference.ncaaf.teams.Team* attribute), 93

`yards_per_play` (*sportsreference.nfl.teams.Team* attribute), 115

`yards_per_punt` (*sportsreference.nfl.player.AbstractPlayer* attribute), 102

`yards_per_punt_return` (*sportsreference.nfl.player.AbstractPlayer* attribute), 102

`yards_per_punt_return` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_per_touch` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_recovered_from_fumble` (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

`yards_recovered_from_fumble` (*sportsreference.ncaaf.roster.Player* attribute), 86

`yards_recovered_from_fumble` (*sportsreference.nfl.player.AbstractPlayer* attribute), 102

`yards_recovered_from_fumble` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_returned_from_interception` (*sportsreference.nfl.player.AbstractPlayer* attribute), 102

`yards_returned_from_interception` (*sportsreference.nfl.roster.Player* attribute), 108

`yards_returned_from_interceptions` (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

`yards_returned_from_interceptions` (*sportsreference.ncaaf.roster.Player* attribute), 87

`yards_returned_per_interception` (*sportsreference.ncaaf.player.AbstractPlayer* attribute), 81

`yards_returned_per_interception` (*sportsreference.ncaaf.roster.Player* attribute), 87

`year` (*sportsreference.ncaaf.roster.Player* attribute), 87