NBA Stats Tracking

Aug 28, 2021

Contents

A package to simplify working with NBA player tracking stats from NBA Advanced Stats.

CHAPTER 1

Features

- Works with both tracking stats and tracking shot stats
- Aggregate stats across multiple seasons
- Aggregate tracking shot stats across multiple filters (ex Wide Open and 18-22 seconds left on the shot clock)
- Generate game logs

CHAPTER 2

Quickstart Guide

2.1 Installation

```
$ pip install nba_stats_tracking
```

2.2 Code Examples

2.2.1 Aggregating Multiple Tracking Shot Stat Filters and/or Seasons

The following will get aggregate player stats for Catch and Shoot, Open or Wide-Open shots in the Regular Season and Playoffs from 2013-14 to 2019-20:

```
from nba_stats_tracking import tracking_shots

seasons = ['2013-14', '2014-15', '2015-16', '2016-17', '2017-18', '2018-19', '2019-20
↪']
season_types = ['Regular Season', 'Playoffs']
def_distances = ['6+ Feet - Wide Open', '4-6 Feet - Open']
general_ranges = ['Catch and Shoot']

stats, league_totals = tracking_shots.aggregate_full_season_tracking_shot_stats_for_
↪seasons(
    'player',
    seasons,
    season_types,
    close_def_dists=def_distances,
    general_ranges=general_ranges
)

for stat in stats:
```

(continues on next page)

(continued from previous page)

```
    print(stat)
print(league_totals)
```

2.2.2 Generating Tracking Shot Game Logs

The following gets player game logs for Open and Wide Open Catch and Shoot shots for games from 02/02/2020 to 02/03/2020:

```
from nba_stats_tracking import tracking_shots

def_distances = ['6+ Feet - Wide Open', '4-6 Feet - Open']
general_ranges = ['Catch and Shoot']
date_from = '02/02/2020'
date_to = '02/03/2020'

game_logs = tracking_shots.generate_tracking_shot_game_logs(
    'player',
    date_from,
    date_to,
    close_def_dists=def_distances,
    general_ranges=general_ranges
)
for game_log in game_logs:
    print(game_log)
```

2.2.3 Aggregating Multiple Tracking Shot Stat Filters and Grouping by Season

The following gets player stats for Catch and Shoot, Open or Wide-Open shots in the Regular Season from 2013-14 to 2019-20 and groups the results by season:

```
from nba_stats_tracking import tracking_shots

seasons = ['2013-14', '2014-15', '2015-16', '2016-17', '2017-18', '2018-19', '2019-20
↪']
season_types = ['Regular Season']
def_distances = ['6+ Feet - Wide Open', '4-6 Feet - Open']
general_ranges = ['Catch and Shoot']

stats = tracking_shots.get_tracking_shot_stats(
    'player',
    seasons,
    season_types,
    close_def_dists=def_distances,
    general_ranges=general_ranges
)

for stat in stats:
    print(stat)
```

2.2.4 Aggregating Multiple Seasons of Tracking Stats

The following gets player speed and distance stats from 2018-19 to 2019-20:

```
from nba_stats_tracking import tracking

stat_measure = 'SpeedDistance'
seasons = ['2018-19', '2019-20']
season_types = ['Regular Season']
entity_type = 'player'
stats, league_totals = tracking.aggregate_full_season_tracking_stats_for_seasons(
    stat_measure,
    seasons,
    season_types,
    entity_type
)

for stat in stats:
    print(stat)

print('-----------------------')
print(league_totals)
```

2.2.5 Generating Tracking Game Logs

The following gets player game logs for catch and shoot shots for games from 02/02/2020 to 02/03/2020:

```
from nba_stats_tracking import tracking

stat_measure = 'CatchShoot'
entity_type = 'player'
date_from = '02/02/2020'
date_to = '02/03/2020'

game_logs = tracking.generate_tracking_game_logs(stat_measure, entity_type, date_from,
↪ date_to)
for game_log in game_logs:
    print(game_log)
```

2.2.6 Get Opponent Tracking Stats For An Individual Team

The following gets opponent catch and shoot stats for the Boston Celtics in 2019-20

```
from nba_stats_tracking import tracking

stat_measure = 'CatchShoot'
seasons = ['2019-20']
season_types = ['Regular Season']
entity_type = 'team'
opponent_team_id = 1610612738

# stats will be each team's stats against opponent_team_id
# league_totals will be aggregate opponent stats for opponents of opponent_team_id
stats, league_totals = tracking.aggregate_full_season_tracking_stats_for_seasons(
    stat_measure,
    seasons,
    season_types,
    entity_type,
```

(continues on next page)

(continued from previous page)

```
    opponent_team_id=opponent_team_id
)

for stat in stats:
    print(stat)
print(league_totals)
```

CHAPTER 3

Documentation

3.1 Modules

3.1.1 tracking

`nba_stats_tracking.tracking.`**`add_to_tracking_totals`**(*totals*, *item*)

Adds totals from item to totals

Parameters

- **`totals`** (*`dict`*) – Totals to be added to
- **`item`** (*`dict`*) – Item to be added to totals dict

Returns totals dict

Return type dict

`nba_stats_tracking.tracking.`**`aggregate_full_season_tracking_stats_for_seasons`**(*stat_measure*, *seasons*, *season_types*, *entity_type*, ***kwargs*)

Aggregates full season stats for stat measure for desired filters. Returns list of dicts for stats for each team/player and dict with league totals.

Parameters

- **`stat_measure`** (*`str`*) – Options: Drives, Defense, CatchShoot, Passing, Possessions, PullUpShot, Rebounding, Efficiency, SpeedDistance, ElbowTouch, PostTouch, PaintTouch
- **`seasons`** (*`list[str]`*) – List of seasons.Format YYYY-YY ex 2019-20

- **season_types** (*list[str]*) – List of season types. Options are Regular Season or Playoffs or Play In
- **entity_type** (*str*) – Options are player or team
- **opponent_team_id** (*str*) – (optional) nba.com team id

Returns tuple with list of dicts for stats for each player/team and dict with league totals

Return type tuple(list[dict], dict)

nba_stats_tracking.tracking.**generate_tracking_game_logs**(*stat_measure*, *entity_type*, *date_from*, *date_to*, ***kwargs*)

Generates game logs for all games between two dates for desired filters

Parameters

- **stat_measure** (*str*) – Options: Drives, Defense, CatchShoot, Passing, Possessions, PullUpShot, Rebounding, Efficiency, SpeedDistance, ElbowTouch, PostTouch, PaintTouch
- **entity_type** (*str*) – Options are player or team
- **date_from** (*str*) – Format - MM/DD/YYYY
- **date_to** (*str*) – Format - MM/DD/YYYY
- **team_id_game_id_map** (*dict*) – (optional) dict mapping team id to game id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times
- **team_id_opponent_team_id_map** (*dict*) – (optional) dict mapping team id to opponent team id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times
- **player_id_team_id_map** (*dict*) – (optional) dict mapping player id to team id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times

Returns list of game log dicts

Return type list[dict]

nba_stats_tracking.tracking.**get_tracking_response_json_for_stat_measure**(*stat_measure*, *season*, *season_type*, *entity_type*, *per_mode*, ***kwargs*)

Makes API call to NBA Advanced Stats and returns JSON response

Parameters

- **stat_measure** (*str*) – Options: Drives, Defense, CatchShoot, Passing, Possessions, PullUpShot, Rebounding, Efficiency, SpeedDistance, ElbowTouch, PostTouch, PaintTouch
- **season** (*str*) – Format YYYY-YY ex 2019-20
- **season_type** (*str*) – Options are Regular Season or Playoffs or Play In
- **entity_type** (*str*) – Options are player or team

- **per_mode** (*str*) – Options are PerGame and Totals
- **date_from** (*str*) – (optional) Format - MM/DD/YYYY
- **date_to** (*str*) – (optional) Format - MM/DD/YYYY
- **opponent_team_id** (*str*) – (optional) nba.com team id

Returns response json

Return type dict

`nba_stats_tracking.tracking.`**`get_tracking_stats`**(*stat_measure*, *seasons*, *season_types*, *entity_type*, *per_mode='Totals'*, ***kwargs*)

Gets stat measure tracking stats for filter

Parameters

- **stat_measure** (*str*) – Options: Drives, Defense, CatchShoot, Passing, Possessions, PullUpShot, Rebounding, Efficiency, SpeedDistance, ElbowTouch, PostTouch, PaintTouch
- **seasons** (*list[str]*) – List of seasons.Format YYYY-YY ex 2019-20
- **season_types** (*list[str]*) – List of season types. Options are Regular Season or Playoffs or Play In
- **entity_type** (*str*) – Options are player or team
- **per_mode** (*str*) – Options are PerGame and Totals. Defaults to totals.
- **date_from** (*str*) – (optional) Format - MM/DD/YYYY
- **date_to** (*str*) – (optional) Format - MM/DD/YYYY
- **opponent_team_id** (*str*) – (optional) nba.com team id

Returns list of dicts with stats for each player/team

Return type list[dict]

`nba_stats_tracking.tracking.`**`sum_tracking_totals`**(*entity_type*, **args*)

Sums totals for given dicts and grouped by entity type

Parameters

- **entity_type** (*str*) – Options are player, team, opponent or league
- ***args** (*dict*) – Variable length argument list of dicts to be summed up

Returns list of dicts with totals for each entity

Return type list[dict]

3.1.2 tracking_shots

Module containing functions for accessing tracking shot stats

`nba_stats_tracking.tracking_shots.`**`add_to_tracking_shot_totals`**(*totals*, *item*)

Adds shot totals from item to totals and updates percentages

Parameters

- **totals** (*dict*) – Totals to be added to
- **item** (*dict*) – Item to be added to totals dict

Returns totals dict

Return type dict

`nba_stats_tracking.tracking_shots.`**`aggregate_full_season_tracking_shot_stats_for_seasons`**(*ent*
sea
son
sea
son
***k*

Aggregates full season stats for desired filters. Returns list of dicts for stats for each team/player and dict with league totals.

Parameters

- **`entity_type`** (*`str`*) – Options are player, team or opponent
- **`seasons`** (*`list[str]`*) – List of seasons.Format YYYY-YY ex 2019-20
- **`season_types`** (*`list[str]`*) – List of season types. Options are Regular Season or Playoffs or Play In
- **`close_def_dists`** (*`list[str]`*) – (optional) Options: ‘’, ‘0-2 Feet - Very Tight’, ‘2-4 Feet - Tight’,’4-6 Feet - Open’,’6+ Feet - Wide Open’
- **`shot_clocks`** (*`list[str]`*) – (optional) - Options: ‘’, ‘24-22’, ‘22-18 Very Early’, ‘18-15 Early’, ‘15-7 Average’, ‘7-4 Late’, ‘4-0 Very Late’
- **`shot_dists`** (*`list[str]`*) – (optional) - Options: ‘’, ‘>=10.0’
- **`touch_times`** (*`list[str]`*) – (optional) - Options: ‘’, ‘Touch < 2 Seconds’, ‘Touch 2-6 Seconds’, ‘Touch 6+ Seconds’
- **`dribble_ranges`** (*`list[str]`*) – (optional) - Options: ‘’, ‘0 Dribbles’, ‘1 Dribble’, ‘2 Dribbles’, ‘3-6 Dribbles’, ‘7+ Dribbles’
- **`general_ranges`** (*`list[str]`*) – (optional) - Options: ‘Overall’, ‘Catch and Shoot’, ‘Pullups’, ‘Less Than 10 ft’
- **`periods`** (*`list[int]`*) – (optional) Only get stats for specific periods
- **`location`** (*`str`*) – (optional) - Options: ‘Home’ or ‘Road’

Returns tuple with list of dicts for stats for each player/team and dict with league totals

Return type tuple(list[dict], dict)

`nba_stats_tracking.tracking_shots.`**`generate_tracking_shot_game_logs`**(*entity_type*, *date_from*, *date_to*, ***kwargs*)

Generates game logs for all games between two dates for desired filters

Parameters

- **`entity_type`** (*`str`*) – Options are player, team or opponent
- **`date_from`** (*`str`*) – Format - MM/DD/YYYY
- **`date_to`** (*`str`*) – Format - MM/DD/YYYY
- **`team_id_game_id_map`** (*`dict`*) – (optional) dict mapping team id to game id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times

- **team_id_opponent_team_id_map** (*dict*) – (optional) dict mapping team id to opponent team id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times
- **player_id_team_id_map** (*dict*) – (optional) dict mapping player id to team id. When getting game logs for multiple separate filters for the same date it is recommended that you pass this in to avoid making the same request multiple times
- **close_def_dists** (*list[str]*) – (optional) Options: '', '0-2 Feet - Very Tight', '2-4 Feet - Tight','4-6 Feet - Open','6+ Feet - Wide Open'
- **shot_clocks** (*list[str]*) – (optional) - Options: '', '24-22', '22-18 Very Early', '18-15 Early', '15-7 Average', '7-4 Late', '4-0 Very Late'
- **shot_dists** (*list[str]*) – (optional) - Options: '', '>=10.0'
- **touch_times** (*list[str]*) – (optional) - Options: '', 'Touch < 2 Seconds', 'Touch 2-6 Seconds', 'Touch 6+ Seconds'
- **dribble_ranges** (*list[str]*) – (optional) - Options: '', '0 Dribbles', '1 Dribble', '2 Dribbles', '3-6 Dribbles', '7+ Dribbles'
- **general_ranges** (*list[str]*) – (optional) - Options: 'Overall', 'Catch and Shoot', 'Pullups', 'Less Than 10 ft'
- **periods** (*list[int]*) – (optional) Only get stats for specific periods
- **location** (*str*) – (optional) - Options: 'Home' or 'Road'

Returns list of game log dicts

Return type list[dict]

`nba_stats_tracking.tracking_shots.`**`get_tracking_shot_stats`**(*entity_type*, *seasons*, *season_types*, ***kwargs*)

Gets tracking shot stats for filters

Parameters

- **entity_type** (*str*) – Options are player, team or opponent
- **seasons** (*list[str]*) – List of seasons.Format YYYY-YY ex 2019-20
- **season_types** (*list[str]*) – List of season types. Options are Regular Season or Playoffs or Play In
- **close_def_dists** (*list[str]*) – (optional) Options: '', '0-2 Feet - Very Tight', '2-4 Feet - Tight','4-6 Feet - Open','6+ Feet - Wide Open'
- **shot_clocks** (*list[str]*) – (optional) - Options: '', '24-22', '22-18 Very Early', '18-15 Early', '15-7 Average', '7-4 Late', '4-0 Very Late'
- **shot_dists** (*list[str]*) – (optional) - Options: '', '>=10.0'
- **touch_times** (*list[str]*) – (optional) - Options: '', 'Touch < 2 Seconds', 'Touch 2-6 Seconds', 'Touch 6+ Seconds'
- **dribble_ranges** (*list[str]*) – (optional) - Options: '', '0 Dribbles', '1 Dribble', '2 Dribbles', '3-6 Dribbles', '7+ Dribbles'
- **general_ranges** (*list[str]*) – (optional) - Options: 'Overall', 'Catch and Shoot', 'Pullups', 'Less Than 10 ft'
- **date_from** (*str*) – (optional) Format - MM/DD/YYYY

- **date_to** (*str*) – (optional) Format - MM/DD/YYYY
- **periods** (*list[int]*) – (optional) Only get stats for specific periods
- **location** (*str*) – (optional) - Options: 'Home' or 'Road'

Returns list of dicts with stats for each player/team

Return type list[dict]

`nba_stats_tracking.tracking_shots.`**`get_tracking_shots_response`**(*entity_type*, *season*, *season_type*, ***kwargs*)

Makes API call to NBA Advanced Stats and returns JSON response

Parameters

- **entity_type** (*str*) – Options are player, team or opponent
- **season** (*str*) – Format YYYY-YY ex 2019-20
- **season_type** (*str*) – Options are Regular Season or Playoffs or Play In
- **date_from** (*str*) – (optional) Format - MM/DD/YYYY
- **date_to** (*str*) – (optional) Format - MM/DD/YYYY
- **close_def_dist** (*str*) – (optional) Defaults to "". Options: '', '0-2 Feet - Very Tight', '2-4 Feet - Tight','4-6 Feet - Open','6+ Feet - Wide Open'
- **shot_clock** (*str*) – (optional) - Defaults to "". Options: '', '24-22', '22-18 Very Early', '18-15 Early', '15-7 Average', '7-4 Late', '4-0 Very Late'
- **shot_dist** (*str*) – (optional) - Defaults to "". Options: '', '>=10.0'
- **touch_time** (*str*) – (optional) - Defaults to "". Options: '', 'Touch < 2 Seconds', 'Touch 2-6 Seconds', 'Touch 6+ Seconds'
- **dribbles** (*str*) – (optional) - Defaults to "". Options: '', '0 Dribbles', '1 Dribble', '2 Dribbles', '3-6 Dribbles', '7+ Dribbles'
- **general_range** (*str*) – (optional) - Defaults to "Overall". Options: 'Overall', 'Catch and Shoot', 'Pullups', 'Less Than 10 ft'
- **period** (*int*) – (optional) Only get stats for specific period
- **location** (*str*) – (optional) - Options: 'Home' or 'Road'

Returns response json

Return type dict

`nba_stats_tracking.tracking_shots.`**`sum_tracking_shot_totals`**(*entity_type*, **args*)

Sums totals for given dicts and grouped by entity type

Parameters

- **entity_type** (*str*) – Options are player, team, opponent or league
- ***args** (*dict*) – Variable length argument list of dicts to be summed up

Returns list of dicts with totals for each entity

Return type list[dict]

3.1.3 utils

`nba_stats_tracking.utils.`**`get_boxscore_response_for_game`**(*game_id*)
Gets response from boxscore endpoint

Parameters **`game_id`** (*`str`*) – nba.com game id

Returns response json

Return type dict

`nba_stats_tracking.utils.`**`get_game_ids_for_date`**(*date*)
Gets game ids for all games played on a given date

Parameters **`date`** (*`str`*) – Format - MM/DD/YYYY

Returns list of game ids

Return type list

`nba_stats_tracking.utils.`**`get_json_response`**(*url*, *params*)
Helper function to get json response for request

Parameters

- **`url`** (*`str`*) – base url for api endpoint
- **`params`** (*`dict`*) – params for request

Returns response json

Return type dict

`nba_stats_tracking.utils.`**`get_player_team_map_for_date`**(*date*)
Creates a dict mapping player id to team id for all games on a given date

Parameters **`date`** (*`str`*) – Format - MM/DD/YYYY

Returns player id team id dict

Return type dict

`nba_stats_tracking.utils.`**`get_scoreboard_response_json_for_date`**(*date*)
Gets response from scoreboard endpoint

Parameters **`date`** (*`str`*) – Format - MM/DD/YYYY

Returns response json

Return type dict

`nba_stats_tracking.utils.`**`get_season_from_game_id`**(*game_id*)
Gets season from nba.com game id 4th and 5th digits of game id represent year season started ex 0021900001 is for the 2019-20 season

Parameters **`game_id`** (*`str`*) – nba.com game id

Returns season - Format YYYY-YY ex 2019-20

Return type string

`nba_stats_tracking.utils.`**`get_season_type_from_game_id`**(*game_id*)
Gets season type from nba.com game id Season type is represented in 3rd digit of game id 2 is Regular Season, 4 is Playoffs

Parameters **`game_id`** (*`str`*) – nba.com game id

Returns season type - Regular Season or Playoffs

Return type string

`nba_stats_tracking.utils.`**`get_team_id_maps_for_date`**(*date*)

Creates dicts mapping team id to game id and team id to opponent team id for games on a given date

Parameters **`date`** (*`str`*) – Format - MM/DD/YYYY

Returns team id game id dict, team id opponent id dict

Return type tuple(dict, dict)

`nba_stats_tracking.utils.`**`make_array_of_dicts_from_response_json`**(*response_json*, *index*)

Makes array of dicts from stats.nba.com response json

Parameters

- **`response_json`** (*`dict`*) – dict with response from request
- **`index`** (*`int`*) – index that holds results in resultSets array

Returns list of dicts with data for each row

Return type list[dict]

`nba_stats_tracking.utils.`**`make_player_team_map_for_game`**(*boxscore_data*)

Creates a dict mapping player id to team id for a game

Parameters **`boxscore_data`** (*`dict`*) – list of dicts with boxscore data for a game

Returns player id team id dict

Return type dict

CHAPTER 4

Notes

It looks like prior to 2018-19 blocked shots aren't included in the FGA tracking shot totals

Python Module Index

Index