IDS xercise 1

Riccardo Cereghino

Mar 18, 2021

CONTENTS:

This module considers the following csv structure to indicate each team with basic values to be used for statistics. .. csv

https://www.kaggle.com/martj42/international-football-results-from-1872-to-2017

It currently generates and graphs the indicators for Iceland and Italy.

Example

The basic usage to generate $ indicators = generate_indicators(os.path.abspath('indicator/results.csv')) ind_1, ind_2, ind_3 = tee(indicators, 3) {

print("Iceland indicators") S = list(select(ind_1, team_name__eq="Iceland")).pop() prettify(S) plot(S)

} search_params = {

"mode": "and", "wins__gt": S.get("wins"), "losses__lt": S.get("losses"), "avg_goals_scored__gt": S.get("avg_goals_scored"), "avg_goals_taken__lt": S.get("avg_goals_taken"), "max_win_streak__gt": S.get("max_win_streak")

}

print("Teams with indicators better than Iceland") prettyficator(select(ind=ind_2, **search_params))

print("Italy indicators") S = list(select(ind_3, team_name__eq="Italy")).pop() prettify(S) plot(S)

$ python example_google.py

Section breaks are created by resuming unindented text. Section breaks are also implicitly created anytime a new section starts.

`indicator.functional.`**`module_level_variable1`**

Module level variables may be documented in either the `Attributes` section of the module docstring, or in an inline docstring immediately following the variable.

Either form is acceptable, but the two should not be mixed. Choose one convention to document module level variables and be consistent with it.

Type int

`indicator.functional.`**`generate_rows`**(*file_name*)

Generators have a `Yields` section instead of a `Returns` section.

Parameters **`n`** (*int*) – The upper limit of the range to generate, from 0 to *n* - 1.

Yields *int* – The next number in the range of 0 to *n* - 1.

Examples

Examples should be written in doctest format, and should illustrate how to use the function.

$ print([i for i in example_generator(4)]) [0, 1, 2, 3]

`indicator.functional.`**`row_splitter`**(*row*)

Example function with types documented in the docstring.

PEP 484 type annotations are supported. If attribute, parameter, and return types are annotated according to PEP 484, they do not need to be included in the docstring:

Parameters

- **`param1`** (*int*) – The first parameter.

- **param2** (*str*) – The second parameter.

Returns The return value. True for success, False otherwise.

Return type bool

indicator.functional.**update_indicator**(*ind*, *md*)

This is an example of a module level function.

Function parameters should be documented in the `Args` section. The name of each parameter is required. The type and description of each parameter is optional, but should be included if not obvious.

If *args or **kwargs are accepted, they should be listed as `*args` and `**kwargs`.

The format for a parameter is:

```
name (type): description
    The description may span multiple lines. Following
    lines should be indented. The "(type)" is optional.

    Multiple paragraphs are supported in parameter
    descriptions.
```

Parameters

- **param1** (*int*) – The first parameter.
- **param2** (`str`, optional) – The second parameter. Defaults to None. Second line of description should be indented.
- ***args** – Variable length argument list.
- ****kwargs** – Arbitrary keyword arguments.

Returns

True if successful, False otherwise.

The return type is optional and may be specified at the beginning of the `Returns` section followed by a colon.

The `Returns` section may span multiple lines and paragraphs. Following lines should be indented to match the first line.

The `Returns` section supports any reStructuredText formatting, including literal blocks:

```
{
    'param1': param1,
    'param2': param2
}
```

Return type bool

Raises

- **AttributeError** – The `Raises` section is a list of all exceptions that are relevant to the interface.
- **ValueError** – If *param2* is equal to *param1*.

PYTHON MODULE INDEX

i

INDEX