

Project

API Documentation

December 9, 2016

Contents

| | |
|---|-----------|
| Contents | 1 |
| 1 Package analisis-machismo | 2 |
| 1.1 Modules | 2 |
| 2 Package analisis-machismo.app | 3 |
| 2.1 Modules | 3 |
| 3 Module analisis-machismo.app.analysis | 4 |
| 3.1 Functions | 4 |
| 4 Module analisis-machismo.app.dictionary_tagger | 5 |
| 4.1 Class DictionaryTagger | 5 |
| 4.1.1 Methods | 5 |
| 5 Module analisis-machismo.app.key_reader | 6 |
| 5.1 Class KeyReader | 6 |
| 5.1.1 Methods | 6 |
| 6 Module analisis-machismo.app.tag_counter | 7 |
| 6.1 Class TagCounter | 7 |
| 6.1.1 Methods | 7 |
| 7 Module analisis-machismo.app.tweet_formatter | 8 |
| 7.1 Class TweetFormatter | 8 |
| 7.1.1 Methods | 8 |
| 8 Module analisis-machismo.app.twitter_miner | 9 |
| 8.1 Class TwitterMiner | 9 |
| 8.1.1 Methods | 9 |
| 9 Package analisis-machismo.tests | 10 |
| 9.1 Modules | 10 |
| 10 Module analisis-machismo.tests.test_dictionary_tagger | 11 |
| 10.1 Class TestDictionaryTagger | 11 |
| 10.1.1 Methods | 11 |

| | |
|---|-----------|
| 11 Module analisis-machismo.tests.test_key_reader | 12 |
| 11.1 Class TestKeyReader | 12 |
| 11.1.1 Methods | 12 |
| 12 Module analisis-machismo.tests.test_tweet_formatter | 13 |
| 12.1 Class TestTweetFormatter | 13 |
| 12.1.1 Methods | 13 |
| Index | 14 |

1 Package analisis-machismo

1.1 Modules

- **app** (*Section 2, p. 3*)
 - **analysis** (*Section 3, p. 4*)
 - **dictionary_tagger** (*Section 4, p. 5*)
 - **key_reader** (*Section 5, p. 6*)
 - **tag_counter** (*Section 6, p. 7*)
 - **tweet_formatter** (*Section 7, p. 8*)
 - **twitter_miner** (*Section 8, p. 9*)
- **tests** (*Section 9, p. 10*)
 - **test_dictionary_tagger** (*Section 10, p. 11*)
 - **test_key_reader** (*Section 11, p. 12*)
 - **test_tweet_formatter** (*Section 12, p. 13*)

2 Package analisis-machismo.app

2.1 Modules

- **analysis** (*Section 3, p. 4*)
- **dictionary_tagger** (*Section 4, p. 5*)
- **key_reader** (*Section 5, p. 6*)
- **tag_counter** (*Section 6, p. 7*)
- **tweet_formatter** (*Section 7, p. 8*)
- **twitter_miner** (*Section 8, p. 9*)

3 Module analisis-machismo.app.analysis

3.1 Functions

```
main(keys='keys.ini', raw_tweets_file='twitter_data.txt', no_tweets=1000,  
tracked_words_file='tracks.csv', formatted_tweets_file='formatted_tweets.txt',  
dictionaries=['misoginy_dictionary.yml', 'curses_dictionary.yml'])
```

Perform an analysis to find sexist and rude words in tweets

This module employs every other module to perform a full analysis on data retrieved from the Twitter stream. First a TwitterMiner retrieves data and dumps it, then a TweetFormatter parses the data into a list of tweets that are lists of words. Then it uses the spaghetti tagger to POSTag every word, yielding a list of tweets that are lists with elements with the form (word, [tags]). A DictionaryTagger adds our custom tags to the [tags] list. Finally a TagCounter perform a count of every tag found in tweets. This program prints the number of coincidences of our custom tags.

4 Module `analisiis-machismo.app.dictionary_tagger`

4.1 Class `DictionaryTagger`

Python class for tagging text with dictionaries

4.1.1 Methods

| |
|---|
| <code>__init__(self, dictionary_paths)</code> |
| Dictionary is a dict containing all the dictionaries parsed from the paths given. |

| |
|--|
| <code>compile_dictionaries(self, dictionary_paths)</code> |
| Returns a list of dictionaries parsed from .yaml files |

| |
|--|
| <code>tag(self, postagged_sentences)</code> |
|--|

| |
|---|
| <code>tag_sentence(self, sentence, tag_with_lemmas=None)</code> |
| The result is only tagging of all the possible ones. The resulting tagging is determined by these two priority rules: |
| <ul style="list-style-type: none">• longest matches have higher priority• search is made left to right |

5 Module analisis-machismo.app.key_reader

5.1 Class KeyReader

Wrapper of ConfigParser to load .ini file with Twitter API keys

5.1.1 Methods

| |
|-----------------------------|
| <code>__init__(self)</code> |
|-----------------------------|

| |
|--|
| <code>read(self, filename='keys.ini', section='keys')</code> |
|--|

| |
|---|
| Import the configparser, tell it to read the file, and get a listing of the sections. Sections are listed in a python dictionary. |
|---|

6 Module *analisiis-machismo.app.tag_counter*

6.1 Class `TagCounter`

Wrapper for a tag counting method

6.1.1 Methods

| |
|--|
| <code>__init__(self, tagged_tweets)</code> |
| Store a list of tweets in case none is provided in count() |

| |
|---|
| <code>count(self, tagged_tweets=None)</code> |
| Handle the counting of tags inserting them in a dictionary to ensure their uniqueness. Can manage a list of tags and a single string tag. |

7 Module *alisis-machismo.app.tweet_formatter*

7.1 Class `TweetFormatter`

Provides the tools needed to format raw data from Twitter stream to stripped text like this: raw -> json -> text -> stripped text.

7.1.1 Methods

| |
|---|
| <code>__init__(self, source_file=None)</code> |
|---|

| |
|---|
| <code>convert2json(self, tweets_source=None)</code> |
|---|

| |
|--|
| Wrap json module to convert raw Twitter data to json |
|--|

| |
|---|
| <code>convert2text(self, tweets_data=None, output_file=None)</code> |
|---|

| |
|---|
| Grab 'text' field of jsons only and return a list of them |
|---|

| |
|---|
| <code>clean_tweets(self, tweets_text=None)</code> |
|---|

| |
|--|
| Regular expressions to remove unnecessary characters in Tweets |
|--|

8 Module `analisi-machismo.app.twitter_miner`

8.1 Class `TwitterMiner`

Exposes the whole chain needed to retrieve data from the Twitter stream from reading the keys, create an `oAuth` object,

8.1.1 Methods

| |
|--|
| <code>__init__(self, keys_file='keys.ini', output_file=None, max_tweets=10)</code> |
|--|

| |
|----------------------------|
| <code>connect(self)</code> |
|----------------------------|

| |
|---|
| Get keys and connect to Twitter through OAuth |
|---|

| |
|--|
| <code>mine(self, track_words, output_file=None)</code> |
|--|

| |
|---|
| Retrieve tweets with text that matches track-words. |
|---|

9 Package analisis-machismo.tests

9.1 Modules

- `test_dictionary_tagger` (*Section 10, p. 11*)
- `test_key_reader` (*Section 11, p. 12*)
- `test_tweet_formatter` (*Section 12, p. 13*)

10 Module `analysis-machismo.tests.test_dictionary_tagger`

10.1 Class `TestDictionaryTagger`

`unittest.TestCase` — `analysis-machismo.tests.test_dictionary_tagger.TestDictionaryTagger`

10.1.1 Methods

| |
|---|
| <code>test_init_output(self)</code> |
| <code>unittest</code> supports test automation, sharing of setup and shutdown code for tests, aggregation of tests into collections, and independence of the tests from the reporting framework. The <code>unittest</code> module provides classes that make it easy to support these qualities for a set of tests. |
| <code>test_tag_sentences(self)</code> |

11 Module `analysis-machismo.tests.test_key_reader`

11.1 Class `TestKeyReader`

`unittest.TestCase` —
`analysis-machismo.tests.test_key_reader.TestKeyReader`

11.1.1 Methods

| |
|---------------------------------|
| <code>setup(self)</code> |
|---------------------------------|

| |
|--|
| <code>test_missing_section(self)</code> |
|--|

| |
|---|
| unittest supports test automation, sharing of setup and shutdown code for tests, aggregation of tests into collections, and independence of the tests from the reporting framework. The unittest module provides classes that make it easy to support these qualities for a set of tests. |
|---|

| |
|--|
| <code>test_valid_files(self)</code> |
|--|

12 Module `analysis-machismo.tests.test_tweet_formatter`

12.1 Class `TestTweetFormatter`

`unittest.TestCase` — `analysis-machismo.tests.test_tweet_formatter.TestTweetFormatter`

The crux of each test is a call to `assertEqual()` to check for an expected result; `assertTrue()` or `assertFalse()` to verify a condition; or `assertRaises()` to verify that a specific exception gets raised. These methods are used instead of the `assert` statement so the test runner can accumulate all test results and produce a report.

12.1.1 Methods

test_inexistent_file(*self*)

test_no_JSON_tweets(*self*)

test_no_text_tweets(*self*)

test_no_text_key(*self*)

Index

- analysis-machismo (*package*), 2
 - analysis-machismo.app (*package*), 3
 - analysis-machismo.app.analysis (*module*), 4
 - analysis-machismo.app.dictionary_tagger (*module*), 5
 - analysis-machismo.app.key_reader (*module*), 6
 - analysis-machismo.app.tag_counter (*module*), 7
 - analysis-machismo.app.tweet_formatter (*module*), 8
 - analysis-machismo.app.twitter_miner (*module*), 9
 - analysis-machismo.tests (*package*), 10
 - analysis-machismo.tests.test_dictionary_tagger (*module*), 11
 - analysis-machismo.tests.test_key_reader (*module*), 12
 - analysis-machismo.tests.test_tweet_formatter (*module*), 13