
pyTweetBot Documentation

Release 0.1

Nils Schaetti

Jul 02, 2018

1	All about me	1
2	Configuration	3
3	Installation	7
4	pyTweetBot.config package	9
5	pyTweetBot.db.obj package	13
6	pyTweetBot.db package	17
7	pyTweetBot.directmessages package	19
8	pyTweetBot.executor package	21
9	pyTweetBot.friends package	23
10	pyTweetBot.learning.features package	25
11	pyTweetBot.learning package	27
12	pyTweetBot.mail package	31
13	pyTweetBot.news package	33
14	pyTweetBot.patterns package	35
15	pyTweetBot.retweet package	37
16	pyTweetBot.stats package	39
17	pyTweetBot.templates package	41
18	pyTweetBot.tools package	43
19	pyTweetBot.tweet package	45
20	pyTweetBot.twitter package	49

21	<code>pyTweetBot.convert_dataset</code>	51
22	<code>pyTweetBot.create_database</code>	53
23	<code>pyTweetBot.direct_messages</code>	55
24	<code>pyTweetBot.execute_actions</code>	57
25	<code>pyTweetBot.export_database</code>	59
26	<code>pyTweetBot.find_follows</code>	61
27	<code>pyTweetBot.find_github_tweets</code>	63
28	<code>pyTweetBot.find_retweets</code>	65
29	<code>pyTweetBot.find_tweets</code>	67
30	<code>pyTweetBot.find_unfollows</code>	69
31	<code>pyTweetBot</code>	71
32	Indices and tables	79
	Python Module Index	81

CHAPTER 1

All about me

I'm Nils Schaetti, a PhD student at the University of Neuchâtel, Switzerland, and developer.

I've contributed to:

- EchoTorch
- TorchLanguage
- pyTweetBot
- pyInstaBot

2.1 Configuration file

pyTweetBot takes its configuration in a JSON file which looks as follow :

```
>>> {
>>>   "database" :
>>>   {
>>>     "host" : "",
>>>     "username" : "",
>>>     "password" : "",
>>>     "database" : ""
>>>   },
>>>   "email" : "bot@bot.com",
>>>   "scheduler" :
>>>   {
>>>     "sleep": [6, 13]
>>>   },
>>>   "hashtags":
>>>   [
>>>   ],
>>>   "twitter" :
>>>   {
>>>     "auth_token2" : "",
>>>     "access_token1" : "",
>>>     "access_token2" : "",
>>>     "user" : ""
>>>   },
>>>   "friends" :
>>>   {
>>>     "max_new_followers" : 40,
>>>     "max_new_unfollow" : 40,
>>>     "follow_unfollow_ratio_limit" : 1.2,
>>>     "interval" : [30, 45]
>>>   },
>>> }
```

(continues on next page)

(continued from previous page)

```

>>> "forbidden_words" :
>>> [
>>> ],
>>> "direct_message" : "",
>>> "tweet" : {
>>>     "max_tweets" : 1200,
>>>     "exclude" : [],
>>>     "interval" : [2.0, 4.0]
>>> },
>>> "news" :
>>> [
>>>     {
>>>         "keyword" : "",
>>>         "countries" : ["us", "fr"],
>>>         "languages" : ["en", "fr"],
>>>         "hashtags" : []
>>>     }
>>> ],
>>> "rss" :
>>> [
>>>     { "url" : "http://feeds.feedburner.com/TechCrunch/startups", "hashtags" : "
↪ #startups", "via" : "@techcrunch" },
>>>     { "url" : "http://feeds.feedburner.com/TechCrunch/fundings-exits",
↪ "hashtags" : "#fundings", "via" : "@techcrunch" }
>>> ],
>>>     "max_retweets" : 600,
>>>     "max_likes" : 600,
>>>     "keywords" : [],
>>>     "nbpages" : 40,
>>>     "retweet_prob" : 0.5,
>>>     "limit_prob" : 1.0
>>>     "interval" : [2.0, 4.0]
>>> },
>>> "github" :
>>> {
>>>     "login": "",
>>>     "password": "",
>>>     "exclude": [],
>>>     "topics" : []
>>> }
>>> }

```

There are two required sections :

- Database : contains the information to connect to the MySQL database (host, username, password, database)
- Twitter : contains the information for the Twitter API (auth and access tokens)

2.2 Database configuration

The database part of the configuration file looks like the following

```

>>> "database" :
>>> {
>>>     "host" : "",

```

(continues on next page)

(continued from previous page)

```
>>> "username" : "",
>>> "password" : "",
>>> "database" : ""
>>> }
```

This section is mandatory.

2.3 Update e-mail configuration

You can configure your bot to send you an email with the number of new followers in the email section

```
>>> "email" : "bot@bot.com"
```

2.4 Scheduler configuration

The scheduler is responsible for executing the bot's actions and you can configure it the sleep for a specific period of time.

```
>>> "scheduler" :
>>> {
>>>     "sleep": [6, 13]
>>> }
```

Here the scheduler will sleep during 6h00 and 13h00.

2.5 Hashtags

You can add text to be replace as hashtags in your tweet in the “hashtags” section

```
>>> "hashtags":
>>> [
>>>     {"from" : "My Hashtag", "to" : "#MyHashtag", "case_sensitive" : true}
>>> ]
```

Here, occurrences of “My Hashtag” will be replaced by #MyHashtag.

2.6 Twitter

To access Twitter, pyTweetBot needs four tokens for the Twitter API and your username.

```
>>> "twitter" :
>>> {
>>>     "auth_token1" : "",
>>>     "auth_token2" : "",
>>>     "access_token1" : "",
>>>     "access_token2" : "",
>>>     "user" : ""
>>> }
```

TODO: tutorial to get the tokens

2.7 Friends settings

The friends section has four parameters.

```
>>> "friends" :
>>> {
>>>     "max_new_followers" : 40,
>>>     "max_new_unfollow" : 40,
>>>     "follow_unfollow_ratio_limit" : 1.2,
>>>     "interval" : [30, 45]
>>> }
```

- The `max_new_followers` set the maximum user that can be followed each day;
- The `max_new_unfollow` set the maximum user that can be unfollowed each day;
- The `interval` parameter set the interval in minutes between each follow/unfollow action choosen randomly between the min and the max;

This note will present an overview of how to install pyTweetBot.

3.1 Getting started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes. See deployment for notes on how to deploy the project on a live system.

3.1.1 Prerequisites

You need to following package to install pyTweetBot.

- nltk
- argparse
- logging
- tweepy
- sklearn
- pygithub
- brotli
- httpplib2
- urlparse2
- HTMLParser
- bs4
- simplejson
- dnspython

- dill
- lxml
- sqlalchemy
- feedparser
- textblob
- numpy
- scipy
- mysql-python

3.2 Installation

```
>>> pip install pyTweetBot
```

3.3 Authors

- **Nils Schaetti** - *Initial work* - (<https://github.com/nschaetti/>)

3.4 License

This project is licensed under the GPLv3 License - see the LICENSE file for details.

4.1 How to use the config package

4.1.1 Required fields

```
>>> required_fields = \
>>> {
>>>     "database":
>>>     {
>>>         "host": {},
>>>         "username": {},
>>>         "password": {},
>>>         "database": {}
>>>     },
>>>     "twitter":
>>>     {
>>>         "auth_token1": {},
>>>         "auth_token2": {},
>>>         "access_token1": {},
>>>         "access_token2": {},
>>>         "user": {}
>>>     }
>>> }
```

4.1.2 Default configuration

```
>>> {
>>>     "database" :
>>>     {
>>>         "host" : "",
>>>         "username" : "",
```

(continues on next page)

(continued from previous page)

```

>>>     "password" : "",
>>>     "database" : ""
>>> },
>>> "email" : "bot@bot.com",
>>> "scheduler" :
>>> {
>>>     "sleep": [6, 13]
>>> },
>>> "hashtags":
>>> [
>>> ],
>>> "twitter" :
>>>     "auth_token2" : "",
>>>     "access_token1" : "",
>>>     "access_token2" : "",
>>>     "user" : ""
>>> },
>>> "friends" :
>>> {
>>>     "max_new_followers" : 40,
>>>     "max_new_unfollow" : 40,
>>>     "follow_unfollow_ratio_limit" : 1.2,
>>>     "interval" : [30, 45]
>>> },
>>> "forbidden_words" :
>>> [
>>> ],
>>> "direct_message" : "",
>>> "tweet" : {
>>>     "max_tweets" : 1200,
>>>     "exclude" : [],
>>>     "interval" : [2.0, 4.0]
>>> },
>>> "news" :
>>> [
>>>     {
>>>         "keyword" : "",
>>>         "countries" : ["us", "fr"],
>>>         "languages" : ["en", "fr"],
>>>         "hashtags" : []
>>>     }
>>> ],
>>> "rss" :
>>> [
>>>     {"url" : "http://feeds.feedburner.com/TechCrunch/startups", "hashtags" : "#startups", "via" : "@techcrunch"},
>>>     {"url" : "http://feeds.feedburner.com/TechCrunch/fundings-exits",
↳ "hashtags" : "#fundings", "via" : "@techcrunch"}
>>> ],
>>>     "max_retweets" : 600,
>>>     "max_likes" : 600,
>>>     "keywords" : [],
>>>     "nbpages" : 40,
>>>     "retweet_prob" : 0.5,
>>>     "limit_prob" : 1.0
>>>     "interval" : [2.0, 4.0]
>>> },

```

(continues on next page)

(continued from previous page)

```

>>>     "github" :
>>>     {
>>>         "login": "",
>>>         "password": "",
>>>         "exclude": [],
>>>         "topics" : []
>>>     }
>>> }

```

4.1.3 Construction

4.2 BotConfig class

class pyTweetBot.config.**BotConfig** (*data*)

This class reads the JSON configuration file and check that all required field is set. It will check that a field is available when asked for or will raise a `FieldNotAvailable` exception.

Arguments: data (dict): Configuration data as a dictionary.

database

Returns: Database configuration (username, password, database)

direct_message

Returns: Direct message configuration (dict)

email

Returns: Email address configuration (dict)

forbidden_words

Returns: Forbidden words configuration (dict)

friends

Returns: Friends configuration (dict)

get_current_interval (*setting*)

Get the interval between actions for the current date and time.

Arguments: setting (dict): The section containing interval data as a dictionary.

Returns: A list (list) with the minimum and maximum time in seconds of the current interval.

get_random_interval (*setting*)

Get a random waiting time for a specific type of actions.

Arguments: setting (str): Setting type. Can be tweet, retweet, like, follow, unfollow

Returns: A time interval as an integer corresponding to the time in seconds.

github

Returns: GitHub configuration (dict)

google_news

Returns: Google News configuration (dict)

hashtags

Returns: Hashtags configuration (dict)

is_available (*key*)

Is a setting available in the loaded configuration?

Arguments: *key* (str): Setting's key in the configuration

is_aware ()

Is the scheduler awake or asleep?

Returns: True if awake, False otherwise

static load (*config_file*)

Load the configuration file

Arguments:

- *config_file* (str): Path to configuration file

Returns: Bot configuration object of type `pyTweetBot.config.BotConfig`.

retweet

Returns: Retweet configuration (dict)

rss

Returns: RSS streams configuration (dict)

scheduler

Returns: Scheduler configuration (dict)

tweet

Returns: Tweet settings configuration (dict)

twitter

Returns: Twitter configuration (dict)

wait_next_action (*setting*)

Wait for a random period corresponding to the current interval of an action type.

Arguments:

- *setting* (dict): Setting type (tweet, retweet, friend) containing an interval field.

5.1 Submodules

5.2 pyTweetBot.db.obj.Action module

```
class pyTweetBot.db.obj.Action.Action(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    action_date
    action_id
    action_order
    action_tweet_id
    action_tweet_text
    action_type
    execute()
        Execute the action :return:
```

5.3 pyTweetBot.db.obj.Base module

5.4 pyTweetBot.db.obj.Follower module

```
class pyTweetBot.db.obj.Follower.Follower(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Friend
    follower_friend
```

```
follower_id
follower_last_update
friend
```

5.5 pyTweetBot.db.obj.Following module

```
class pyTweetBot.db.obj.Following.Following(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base

    follower_followed_date
    following_friend
    following_id
    following_last_update
    friend
```

5.6 pyTweetBot.db.obj.Friend module

```
class pyTweetBot.db.obj.Friend.Friend(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base

    Friend (follower/following) in the database

    follower
        Is the friend a follower? :return: True if follower, False otherwise

    following
        Is the friend a following :return: True if following, False otherwise

    friend_contacted
    friend_description
    friend_follower
    friend_follower_date
    friend_followers_count
    friend_following
    friend_following_date
    friend_friends_count
    friend_id
    friend_last_update
    friend_location
    friend_screen_name
    friend_special
    friend_statuses_count
```

```
static get_friend (name_or_id)  
    Get a friend by it's screen name :param name_or_id: :return:
```

5.7 pyTweetBot.db.obj.ImpactStatistics module

```
class pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic (**kwargs)  
    Bases: sqlalchemy.ext.declarative.api.Base  
  
    Bot's impact statistics  
  
    static exists (week_day, hour)  
        Impact statistics exists? :param week_day: :param hour: :return:  
  
    impact_statistic_count  
  
    impact_statistic_hour  
  
    impact_statistic_id  
  
    impact_statistic_week_day  
  
    static update (week_day, hour, count)  
        Update :param week_day: :param hour: :param count: :return:
```

5.8 pyTweetBot.db.obj.Model module

```
class pyTweetBot.db.obj.Model.Model (**kwargs)  
    Bases: sqlalchemy.ext.declarative.api.Base  
  
    Model description  
  
    static exists (name)  
        Does a model exists? :param name: Model's name :return: True or False  
  
    static get_by_name (name)  
        Get a model by its name :param name: Model's name :return: Model DB object  
  
    model_id  
  
    model_last_update  
  
    model_n_classes  
  
    model_name
```

5.9 pyTweetBot.db.obj.ModelTokens module

```
class pyTweetBot.db.obj.ModelTokens.ModelToken (**kwargs)  
    Bases: sqlalchemy.ext.declarative.api.Base  
  
    Model's tokens  
  
    static get_tokens (model, c=None)  
        Get token probs for a model :param model: Model's name :param c: Class :return:  
  
    model  
  
    token_class
```

```
token_count
token_id
token_model
token_text
token_total
```

5.10 pyTweetBot.db.obj.Statistic module

```
class pyTweetBot.db.obj.Statistic.Statistic(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Bot's statistics

    statistic_date
    statistic_followers_count
    statistic_friends_count
    statistic_id
    statistic_statuses_count
```

5.11 pyTweetBot.db.obj.Tweeted module

```
class pyTweetBot.db.obj.Tweeted.Tweeted(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Tweet

    static exists(tweet)
        Tweet exists :param tweet: :return:

    static insert_retweet(tweet_id, tweet_text)
        Insert a new retweeted :param tweet_id: Tweet's ID :param tweet_text: Tweet's text

    static insert_tweet(tweet_text)
        Insert a new tweeted :param tweet_text: Tweet's text :return:

    tweet_date
    tweet_id
    tweet_tweet_id
    tweet_tweet_text
```

5.12 Module contents

6.1 Subpackages

6.2 Submodules

6.3 pyTweetBot.db.DBConnector module

6.4 Module contents

pyTweetBot.directmessages package

7.1 Submodules

7.2 pyTweetBot.directmessages.directmessages module

```
pyTweetBot.directmessages.directmessages.sendDirectMessage (api, follower,  
                                                             json_data)  
pyTweetBot.directmessages.directmessages.updateFollowers (api, con, user, day_num,  
                                                             json_data)
```

7.3 pyTweetBot.directmessages.pyTweetBotDirectMessageAction module

7.4 pyTweetBot.directmessages.pyTweetBotDirectMessenger module

```
class pyTweetBot.directmessages.pyTweetBotDirectMessenger.pyTweetBotDirectmessenger  
    Bases: object
```

7.5 Module contents

8.1 Submodules

8.2 pyTweetBot.executor.ActionScheduler module

exception pyTweetBot.executor.ActionScheduler.**ActionAlreadyExists**

Bases: exceptions.Exception

The action is already registered in the DB

exception pyTweetBot.executor.ActionScheduler.**ActionReservoirFullError**

Bases: exceptions.Exception

Reservoir is full

exception pyTweetBot.executor.ActionScheduler.**NoFactory**

Bases: exceptions.Exception

No factory to create Tweets

8.3 pyTweetBot.executor.ExecutorThread module

class pyTweetBot.executor.ExecutorThread.**ExecutorThread**(*config*, *scheduler*, *action_type*, *run_event*)

Bases: threading.Thread

Execute actions in a thread

run()

Thread running function :return:

8.4 Module contents

9.1 Submodules

9.2 pyTweetBot.friends.FriendsManager module

exception `pyTweetBot.friends.FriendsManager.ActionAlreadyDone`

Bases: `exceptions.Exception`

Exception, useless action because already done (already following a user)

9.3 Module contents

pyTweetBot.learning.features package

10.1 Submodules

10.2 pyTweetBot.learning.features.BagOf2Grams module

10.3 pyTweetBot.learning.features.BagOf3Grams module

10.4 pyTweetBot.learning.features.BagOfGrams module

10.5 pyTweetBot.learning.features.BagOfWords module

10.6 Module contents

11.1 Subpackages

11.2 Submodules

11.3 pyTweetBot.learning.CensorModel module

```
class pyTweetBot.learning.CensorModel.CensorModel (config)
    Bases: object
    Forbidden words classifier

    static load_censor (config)
        Load a complete model and censor with path to model :param config: :return:
```

11.4 pyTweetBot.learning.Classifier module

11.5 pyTweetBot.learning.Dataset module

```
class pyTweetBot.learning.Dataset.Dataset
    Bases: object
    A dataset of URL and title for training

    add_negative (text)
        Add a positive sample :param text: :return:

    add_positive (text)
        Add a positive sample :param text: :return:
```

```
data
    Data :return:

get_texts ()
    Get texts :return:

is_in (ttext)
    Is in dataset? :param ttext: :return:

static load (opt)
    Load the model from DB or file :param opt: Loading option :return: The model class

next ()
    Next element :return:

save (filename)
    Save the dataset :param filename:

targets
    Targets :return:

to_json ()
    To JSON :return:
```

11.6 pyTweetBot.learning.DecisionTree module

11.7 pyTweetBot.learning.Model module

11.8 pyTweetBot.learning.NaiveBayesClassifier module

11.9 Module contents

```
class pyTweetBot.learning.CensorModel (config)
    Bases: object

    Forbidden words classifier

    static load_censor (config)
        Load a complete model and censor with path to model :param config: :return:

class pyTweetBot.learning.Dataset
    Bases: object

    A dataset of URL and title for training

    add_negative (text)
        Add a positive sample :param text: :return:

    add_positive (text)
        Add a positive sample :param text: :return:

    data
        Data :return:

    get_texts ()
        Get texts :return:
```

is_in (*ttext*)
Is in dataset? :param ttext: :return:

static load (*opt*)
Load the model from DB or file :param opt: Loading option :return: The model class

next ()
Next element :return:

save (*filename*)
Save the dataset :param filename:

targets
Targets :return:

to_json ()
To JSON :return:

12.1 Submodules

12.2 pyTweetBot.mail.MailBuilder module

```
class pyTweetBot.mail.MailBuilder.MailBuilder (message_model)  
    Bases: object  
    Mail builder tool  
    message ()  
        Get message :return: Message as HTML code
```

12.3 pyTweetBot.mail.MailSender module

```
class pyTweetBot.mail.MailSender.MailSender (subject="", from_address="",  
                                              to_addresses="", msg="")  
    Bases: object  
    Mail sender tool  
    from_address (from_address)  
        Set source address :param from_address: :return:  
    send ()  
        Send mail :return: True if ok, False otherwise  
    subject (subject)  
        Set subject :param subject:  
    to_addresses (to_addresses)  
        Set destination addresses :param to_addresses: :return:
```

12.4 Module contents

13.1 Submodules

13.2 pyTweetBot.news.GoogleNewsClient module

class pyTweetBot.news.GoogleNewsClient.**GoogleNewsClient** (*keyword, lang, country*)
Bases: object

This is a Google News client. Which returns an array containing the URLs and titles.

get_news (*page=0*)
Get news :param page: Page to get :return: Array of news

get_page_title (*url*)
Get page's title :param url: :return:

13.3 pyTweetBot.news.NewsParser module

class pyTweetBot.news.NewsParser.**NewsParser**
Bases: HTMLParser.HTMLParser

This is a class parsing HTML from Google news. It returns an array containing the URLs.

get_news ()
Get the news :return:

handle_starttag (*tag, attrs*)
Handle startag :param tag: Tag to handle :param attrs: Tag's attributes

13.4 Module contents

14.1 Submodules

14.2 pyTweetBot.patterns.singleton module

`pyTweetBot.patterns.singleton.singleton(class_)`
Singleton design pattern :param **class_**: :return:

14.3 Module contents

15.1 Submodules

15.2 pyTweetBot.retweet.RetweetFinder module

```
class pyTweetBot.retweet.RetweetFinder.RetweetFinder(search_keywords="",  
                                                    n_pages=-1,    polarity=0.0,  
                                                    subjectivity=0.5,    lan-  
                                                    guages=['en'])
```

Bases: object

Class to find tweet to retweet

next ()

Next element :return:

15.3 Module contents

16.1 Submodules

16.2 pyTweetBot.stats.TweetStatistics module

exception `pyTweetBot.stats.TweetStatistics.TweetAlreadyCountedException`
Bases: `exceptions.Exception`
Exception: the tweet is already counted in stats

class `pyTweetBot.stats.TweetStatistics.TweetStatistics` (*slope=25, beta=5*)
Bases: `object`
TWet statistics managing class

add (*tweet*)
Add a tweet to the stats :param tweet: :return:

count (*weekday, hour*)
Get total counts for a tuple (weekday, hour) :param weekday: :param hour: :return:

expect (*weekday, hour*)
Get expected retweet for a tuple weekday, hour. :param weekday: :param hour: :return:

expect_norm (*weekday, hour*)
Get expected normalized retweet value for a tuple week, hour :param weekday: :param hour: :return:

static load (*filename*)
Load the object :param filename: :return:

save (*filename*)
Save the object to a file :param filename: :return:

start ()
Start statistic counting

stop ()

Stop statistic counting

value (*weekday, hour*)

Get total retweets/likes to a tuple weekday, hour :param weekday: :param hour: :return:

16.3 pyTweetBot.stats.UserStatistics module

16.4 Module contents

CHAPTER 17

pyTweetBot.templates package

17.1 Module contents

18.1 Submodules

18.2 pyTweetBot.tools.PageParser module

class pyTweetBot.tools.PageParser.**PageParser** (*url*, *timeout=20*)

Bases: object

This is a class to retrieve text from HTML page given an URL.

html

Get HTML :return:

raw_title

Raw title :return:

reload (*url=u*)

Reload URL

text

Get text :return:

title

Page's title :return:

url

Loaded URL :return:

exception pyTweetBot.tools.PageParser.**PageParserRetrievalError**

Bases: exceptions.Exception

exception pyTweetBot.tools.PageParser.**UnknownEncoding**

Bases: exceptions.Exception

Unknown encoding exception

18.3 pyTweetBot.tools.strings module

18.4 Module contents

19.1 Submodules

19.2 pyTweetBot.tweet.GoogleNewsHunter module

class pyTweetBot.tweet.GoogleNewsHunter.**GoogleNewsHunter** (*search_term, lang, country, hashtags, languages, n_pages=2*)

Bases: *pyTweetBot.tweet.Hunter.Hunter*

An hunter for Google News

next ()

Next element

Returns: The next tweet

19.3 pyTweetBot.tweet.Hunter module

class pyTweetBot.tweet.Hunter.**Hunter**

Bases: object

next ()

19.4 pyTweetBot.tweet.RSSHunter module

class pyTweetBot.tweet.RSSHunter.**RSSHunter** (*stream*)

Bases: *pyTweetBot.tweet.Hunter.Hunter*

Find new tweets from RSS streams

```
get_stream()  
    Get stream  
  
next()  
    Next :return:
```

19.5 pyTweetBot.tweet.Tweet module

```
class pyTweetBot.tweet.Tweet.Tweet (text, url, hashtags=None)  
    Bases: object  
  
    MAX_LENGTH = 280  
  
    already_tweeted()  
        Already tweeted? :return: True/False  
  
    get_length()  
        Get Tweet length :return:  
  
    get_text()  
        Get Tweet's text. :return: Tweet's text.  
  
    get_tweet()  
        Get Tweet :return: Complete Tweet's text  
  
    get_url()  
        Get Tweet's URL :return: Tweet's URL  
  
    set_text (text)  
        Set Tweet's text :param text: :return:  
  
    set_url (url)  
        Set Tweet's URL :param url: :return:
```

19.6 pyTweetBot.tweet.TweetFactory module

19.7 pyTweetBot.tweet.TweetFinder module

```
class pyTweetBot.tweet.TweetFinder.TweetFinder (shuffle=False, tweet_factory=None)  
    Bases: pyTweetBot.tweet.Hunter.Hunter  
  
    Find new tweets from a set of sources (Google News, RSS)  
  
    add (hunter)  
        Add an hunter to the list :param hunter: The hunter object to add.  
  
    next()  
        Next tweet. :return: The next found tweet.  
  
    next_source()  
        Go to next source  
  
    remove (hunter)  
        Remove hunter :param hunter: The hunter object to remove.  
  
    set_factory (tweet_factory)  
        Set the tweet factory :param tweet_factory: The tweet factory
```

19.8 pyTweetBot.tweet.TweetPreparator module

class pyTweetBot.tweet.TweetPreparator.**TweetPreparator** (*hashtags=None*)
Bases: object
Tweet preparator

19.9 pyTweetBot.tweet.TwitterHunter module

class pyTweetBot.tweet.TwitterHunter.**TwitterHunter** (*search_term,* *hashtags,*
n_pages=2, polarity=0.0, subjec-
tivity=0.5, languages=['en'])

Bases: *pyTweetBot.tweet.Hunter.Hunter*

This class of hunter will find new tweets by scanning URLs in other user's tweets found in research results.

get_hashtags ()
Get hashtags

next ()
Next :return: The next tweet found.

19.10 Module contents

20.1 Submodules

20.2 pyTweetBot.twitter.TweetBotConnect module

exception `pyTweetBot.twitter.TweetBotConnect.RequestLimitReached`

Bases: `exceptions.Exception`

Exception raised when some limits are reached.

20.3 Module contents

CHAPTER 21

pyTweetBot.convert_dataset

This file contains a command line tool to convert a dataset from the old format to the new one. The old format is composed of two lists of URLs and texts. The new dataset format is a Dataset object containing texts and class labels. This tool will download all the page's text of the URIs contained in the old dataset.

Example: Here is a simple example to convert a file:

```
$ python convert_dataset.py --input old.p --output new.p
```

pyTweetBot.create_database

This file contains a function to create the database structure and tables.

Example: Here is a simple example to create the database:

```
>>> config = BotConfig.load("config.json")
>>> create_database(config)
```

22.1 pyTweetBot.create_database module

`pyTweetBot.create_database.create_database(config)`

Function to create the database structure and tables.

Arguments: `config` (BotConfig): The bot configuration object

23.1 `pyTweetBot.direct_messages` module

`pyTweetBot.direct_messages.direct_messages` (*config*)

This function send direct messages to followers if they have not been contacted before.

Example:

```
>>> config = BotConfig.load("config.json")
>>> direct_messages(config)
```

Arguments: `config` (`BotConfig`): Bot configuration object of type `pyTweetBot.config.BotConfig`

pyTweetBot.execute_actions

This file contains a function to launch a thread for each action type that will execute the action accordingly to action scheduler rules.

24.1 pyTweetBot.execute_actions module

```
pyTweetBot.execute_actions.execute_actions(config, action_scheduler, no_tweet=False,  
                                           no_retweet=False, no_like=False,  
                                           no_follow=False, no_unfollow=False)
```

Launch threads that will execute each action thread.

Examples:

```
>>> config = BotConfig.load("config.json")  
>>> action_scheduler = ActionScheduler(config=config)  
>>> execute_actions(config, action_scheduler)
```

Arguments:

- config (BotConfig): Bot configuration of type *pyTweetBot.config.BotConfig*.
- action_scheduler (ActionScheduler): Action management of type *pyTweetBot.executor.ActionScheduler*
- no_tweet (Boolean): Do not execute tweet action
- no_retweet (Boolean): Do not execute retweet action
- no_like (Boolean): Do not execute like action
- no_follow (Boolean): Do not execute follow action
- no_unfollow (Boolean): Do not execute unfollow action

pyTweetBot.export_database

Export a database from a MySQL database to a series of files.

25.1 pyTweetBot.export_database module

`pyTweetBot.export_database.export_database(output_dir, mysql_connector)`

Export a database from a MySQL database to a series of files.

Example:

```
>>> mysql_connector = DBConnector(host="localhost", username="test", password=
↳ "pass", db_name="pytb")
>>> export_database(".", mysql_connector)
```

Arguments:

- `output_dir` (str): The output directory path
- `mysql_connector` (DBConnector) : A connector object of type `pyTweetBot.db.DBConnector`

Find Twitter user to follows accordingly to parameters set in the config file.

26.1 pyTweetBot.find_follows module

`pyTweetBot.find_follows.add_follow_action(action_scheduler, friend)`
Add a follow action through the scheduler.

Arguments:

- `action_scheduler` (`ActionScheduler`): An action scheduler objet of type `pyTweetBot.executor.ActionScheduler`
- `friend` (`Friend` of `tweepy.User`): A friend object (`pyTweetBot.db.obj.Friend`) or a `tweepy.User` object.

`pyTweetBot.find_follows.find_follows(config, model, action_scheduler, friends_manager, text_size, n_pages=20, threshold=0.5)`

Find Twitter user to follows accordingly to parameters set in the config file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> find_follows(config, model, action_scheduler, friends_manager, 50)
```

Arguments:

- `config`: Bot's configuration object
- `model`: Classification model's file
- `action_scheduler`: Action scheduler object
- `friends_manager`: Friends manager object
- `text_size`: Minimum text size to be accepted
- `n_pages`: Number of pages to search for each term

- threshold: Minimum probability to accept following

pyTweetBot.find_github_tweets

Tweet activities of the repositories of an GitHub account like creation and how many pushes. The tweet will look like this :

I made {n} contributions on {date} to project #{project name}, #GitHub #{project topics}

27.1 pyTweetBot.find_github_tweets module

`pyTweetBot.find_github_tweets.add_tweet` (*action_scheduler*, *tweet_text*)
Add tweet through the scheduler

Arguments:

- *action_scheduler*: The action scheduler object
- *tweet_text*: Text to tweet

Returns:

- True if ok, False if problem.

`pyTweetBot.find_github_tweets.compute_tweet` (*tweet_text*, *action_scheduler*, *instantaneous*)
Tweet something directly or add it to the database.

Arguments:

- *tweet_text* (unicode): The text to tweet.
- *action_scheduler* (ActionScheduler): Action scheduler object of type (*pyTweetBot.executor.ActionScheduler*)
- *instantaneous* (bool): Tweet directly (True) or add it to the DB.

Returns:

- True if tweeted/added, False if already in the database.

```
pyTweetBot.find_github_tweets.create_tweet_text (contrib_counter, contrib_date,  
                                                project_name, project_url, topics)
```

Create the tweet's text for a git push event.

Arguments:

- contrib_counter (int): Number of contributions
- contrib_date (datetime): Date of the push
- project_name (unicode): GitHub project's name
- project_url (str): GitHub project's URL
- topics (list): GitHub project's topics

Returns: The tweet's text.

```
pyTweetBot.find_github_tweets.create_tweet_text_create (project_name,  
                                                        project_description,  
                                                        project_url, topics)
```

Create tweet's text for a git repository creation.

Arguments:

- project_name (unicode): GitHub project's name
- project_description (unicode): GitHub project's description
- project_url (unicode): GitHub project's URL
- topics (list): GitHub project's topics.

Returns:

return The created text.

```
pyTweetBot.find_github_tweets.find_github_tweets (config, action_scheduler,  
                                                    event_type='push', depth=-1, in-  
                                                    stantaneous=False, waiting_time=0)
```

Add tweets based on GitHub activities to the database, or tweet it directly.

Arguments:

- config (BotConfig): Bot config object of type `pyTweetBot.config.BotConfig`
- action_scheduler (ActonScheduler): Action scheduler object of type `pyTweetBot.executor.ActionScheduler`
- event_type (str): Type of event to tweet (push or create)
- depth (int): Number of events to tweet for each repository.
- instantaneous: Tweet the information instantaneously or not (to DB)?
- waiting_time: Waiting time between each tweets (for instantaneous tweeting)

```
pyTweetBot.find_github_tweets.prepare_project_name (project_name)
```

Replace - by space in the project name and put the first letter of each word to uppercase.

Arguments:

- project_name (unicode): GitHub project's name

Returns: The cleaned project name

pyTweetBot.find_retweets

Find tweets to retweet accordingly to parameters set in the config file.

28.1 pyTweetBot.find_retweets module

`pyTweetBot.find_retweets.find_retweets`(*config*, *model_file*, *action_scheduler*, *text_size*=80, *threshold*=0.5)

Find tweets to retweet from search terms set in the config file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> find_retweets(config, "model.p", action_scheduler)
```

Arguments:

- *config* (BotConfig): Bot configuration object of type `pyTweetBot.config.BotConfig`
- *model_file* (str): Path to the file containing the classifier model
- *action_scheduler* (ActionScheduler): Action scheduler object of type `pyTweetBot.executor.ActionScheduler`
- *text_size* (int): Minimum text length to take a tweet into account
- *threshold* (float): Minimum to reach to be classified as positive

pyTweetBot.find_tweets

Find tweet from Google News and RSS streams.

29.1 pyTweetBot.find_tweets module

`pyTweetBot.find_tweets.find_tweets` (*config*, *model_file*, *action_scheduler*, *n_pages*=2, *threshold*=0.5)

Find tweet from Google News and RSS streams.

Examples:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> find_tweets(config, "model.p", action_scheduler)
```

Arguments:

- *config* (BotConfig): BotConfig configuration object of type `pyTweetBot.config.BotConfig`
- *model_file* (str): Path to model file for classification
- *action_scheduler* (ActionScheduler): Scheduler object of type `pyTweetBot.executor.ActionScheduler`
- *n_pages* (int): Number of pages to analyze
- *threshold* (float): Probability threshold to be accepted as tweet

pyTweetBot.find_unfollows

Find Twitter users to unfollow according to the parameters in the configuration file.

30.1 pyTweetBot.find_unfollows module

`pyTweetBot.find_unfollows.find_unfollows`(*config*, *friends_manager*, *model_file*, *action_scheduler*, *threshold=0.5*)
Find Twitter users to unfollow according to the parameters in the configuration file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> friends_manager = FriendsManager()
>>> find_unfollows(config, friends_manager, "model.p", action_scheduler)
```

Arguments:

- *config* (BotConfig): Bot configuration object of type `pyTweetBot.config.BotConfig`
- *friends_manager* (FriendsManager): Friend manager object of type `pyTweetBot.friends.FriendsManager`
- *model_file* (str): Path to the model's Pickle file.
- *action_scheduler* (ActionScheduler): Action scheduler object.
- *threshold* (float): Probability threshold to accept unfollow.

31.1 pyTweetBot submodules

31.1.1 Submodules

31.1.2 pyTweetBot.execute_actions module

`pyTweetBot.execute_actions.execute_actions` (*config*, *action_scheduler*, *no_tweet=False*,
no_retweet=False, *no_like=False*,
no_follow=False, *no_unfollow=False*)

Launch threads that will execute each action thread.

Examples:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> execute_actions(config, action_scheduler)
```

Arguments:

- *config* (BotConfig): Bot configuration of type `pyTweetBot.config.BotConfig`.
- *action_scheduler* (ActionScheduler): Action management of type `pyTweetBot.executor.ActionScheduler`
- *no_tweet* (Boolean): Do not execute tweet action
- *no_retweet* (Boolean): Do not execute retweet action
- *no_like* (Boolean): Do not execute like action
- *no_follow* (Boolean): Do not execute follow action
- *no_unfollow* (Boolean): Do not execute unfollow action

31.1.3 pyTweetBot.export_database module

`pyTweetBot.export_database.export_database(output_dir, mysql_connector)`

Export a database from a MySQL database to a series of files.

Example:

```
>>> mysql_connector = DBConnector(host="localhost", username="test", password=
↳ "pass", db_name="pytb")
>>> export_database(".", mysql_connector)
```

Arguments:

- `output_dir` (str): The output directory path
- `mysql_connector` (DBConnector): A connector object of type `pyTweetBot.db.DBConnector`

31.1.4 pyTweetBot.find_follows module

`pyTweetBot.find_follows.add_follow_action(action_scheduler, friend)`

Add a follow action through the scheduler.

Arguments:

- `action_scheduler` (ActionScheduler): An action scheduler object of type `pyTweetBot.executor.ActionScheduler`
- `friend` (Friend of tweepy.User): A friend object (`pyTweetBot.db.obj.Friend`) or a `tweepy.User` object.

`pyTweetBot.find_follows.find_follows(config, model, action_scheduler, friends_manager, text_size, n_pages=20, threshold=0.5)`

Find Twitter user to follows accordingly to parameters set in the config file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> find_follows(config, model, action_scheduler, friends_manager, 50)
```

Arguments:

- `config`: Bot's configuration object
- `model`: Classification model's file
- `action_scheduler`: Action scheduler object
- `friends_manager`: Friends manager object
- `text_size`: Minimum text size to be accepted
- `n_pages`: Number of pages to search for each term
- `threshold`: Minimum probability to accept following

31.1.5 pyTweetBot.find_github_tweets module

`pyTweetBot.find_github_tweets.add_tweet(action_scheduler, tweet_text)`

Add tweet through the scheduler

Arguments:

- `action_scheduler`: The action scheduler object
- `tweet_text`: Text to tweet

Returns:

- True if ok, False if problem.

`pyTweetBot.find_github_tweets.compute_tweet(tweet_text, action_scheduler, instantaneous)`
Tweet something directly or add it to the database.

Arguments:

- `tweet_text` (unicode): The text to tweet.
- `action_scheduler` (ActionScheduler): Action scheduler object of type (`pyTweetBot.executor.ActionScheduler`)
- `instantaneous` (bool): Tweet directly (True) or add it to the DB.

Returns:

- True if tweeted/added, False if already in the database.

`pyTweetBot.find_github_tweets.create_tweet_text(contrib_counter, contrib_date, project_name, project_url, topics)`
Create the tweet's text for a git push event.

Arguments:

- `contrib_counter` (int): Number of contributions
- `contrib_date` (datetime): Date of the push
- `project_name` (unicode): GitHub project's name
- `project_url` (str): GitHub project's URL
- `topics` (list): GitHub project's topics

Returns: The tweet's text.

`pyTweetBot.find_github_tweets.create_tweet_text_create(project_name, project_description, project_url, topics)`
Create tweet's text for a git repository creation.

Arguments:

- `project_name` (unicode): GitHub project's name
- `project_description` (unicode): GitHub project's description
- `project_url` (unicode): GitHub project's URL
- `topics` (list): GitHub project's topics.

Returns:

return The created text.

`pyTweetBot.find_github_tweets.find_github_tweets(config, action_scheduler, event_type='push', depth=-1, instantaneous=False, waiting_time=0)`
Add tweets based on GitHub activities to the database, or tweet it directly.

Arguments:

- `config` (BotConfig): Bot config object of type `pyTweetBot.config.BotConfig`

- `action_scheduler` (ActonScheduler): Action scheduler object of type `pyTweetBot.executor.ActionScheduler`
- `event_type` (str): Type of event to tweet (push or create)
- `depth` (int): Number of events to tweet for each repository.
- `instantaneous`: Tweet the information instantaneously or not (to DB)?
- `waiting_time`: Waiting time between each tweets (for instantaneous tweeting)

`pyTweetBot.find_github_tweets.prepare_project_name` (*project_name*)
Replace - by space in the project name and put the first letter of each word to uppercase.

Arguments:

- `project_name` (unicode): GitHub project's name

Returns: The cleaned project name

31.1.6 pyTweetBot.find_retweets module

`pyTweetBot.find_retweets.find_retweets` (*config, model_file, action_scheduler, text_size=80, threshold=0.5*)
Find tweets to retweet from search terms set in the config file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> find_retweets(config, "model.p", action_scheduler)
```

Arguments:

- `config` (BotConfig): Bot configuration object of type `pyTweetBot.config.BotConfig`
- `model_file` (str): Path to the file containing the classifier model
- `action_scheduler` (ActionScheduler): Action scheduler object of type `pyTweetBot.executor.ActionScheduler`
- `text_size` (int): Minimum text length to take a tweet into account
- `threshold` (float): Minimum to reach to be classified as positive

31.1.7 pyTweetBot.find_tweets module

`pyTweetBot.find_tweets.find_tweets` (*config, model_file, action_scheduler, n_pages=2, threshold=0.5*)
Find tweet from Google News and RSS streams.

Examples:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> find_tweets(config, "model.p", action_scheduler)
```

Arguments:

- `config` (BotConfig): BotConfig configuration object of type `pyTweetBot.config.BotConfig`
- `model_file` (str): Path to model file for classification

- `action_scheduler` (`ActionScheduler`): Scheduler object of type `pyTweetBot.executor.ActionScheduler`
- `n_pages` (int): Number of pages to analyze
- `threshold` (float): Probability threshold to be accepted as tweet

31.1.8 pyTweetBot.find_unfollows module

`pyTweetBot.find_unfollows.find_unfollows` (`config`, `friends_manager`, `model_file`, `action_scheduler`, `threshold=0.5`)

Find Twitter users to unfollow according to the parameters in the configuration file.

Example:

```
>>> config = BotConfig.load("config.json")
>>> action_scheduler = ActionScheduler(config=config)
>>> friends_manager = FriendsManager()
>>> find_unfollows(config, friends_manager, "model.p", action_scheduler)
```

Arguments:

- `config` (`BotConfig`): Bot configuration object of type `pyTweetBot.config.BotConfig`
- `friends_manager` (`FriendsManager`): Friend manager object of type `pyTweetBot.friends.FriendsManager`
- `model_file` (str): Path to the model's Pickle file.
- `action_scheduler` (`ActionScheduler`): Action scheduler object.
- `threshold` (float): Probability threshold to accept unfollow.

31.1.9 pyTweetBot.follower_dataset module

`pyTweetBot.follower_dataset.follower_dataset` (`twitter_connect`, `dataset_file`, `info`, `source='followers'`, `text_size=50`)

Create a dataset or add textual data from a list of Twitter users.

Example:

```
>>> config = BotConfig.load("config.json")
>>> twitter_connector = TweetBotConnector(config)
>>> follower_dataset(twitter_connect, "dataset.p", False, 'followers')
```

Arguments:

- `twitter_connect` (`TweetBotConnector`): Twitter bot connector object of type `pyTweetBot.twitter.TweetBotConnector`
- `dataset_file` (str): Path to the dataset file to load or create.
- `info` (bool): If True, show information about the dataset and exit
- `source` (str): Can be 'follower' or 'following'. Set where to load users from.
- `text_size` (int): Minimum user's description length to take the profile into account.

31.1.10 pyTweetBot.import_database module

`pyTweetBot.import_database.import_actions(session, actions)`
Import actions :param session: :param actions: :return:

`pyTweetBot.import_database.import_database(output_dir, mysql_connector)`
Function to import the database :param output_dir: :param mysql_connector: :return:

`pyTweetBot.import_database.import_friends(session, friends)`
Import friends :param session: :param friends: :return:

`pyTweetBot.import_database.import_statistics(session, statistics)`
Import statistics :param session: :param statistics: :return:

`pyTweetBot.import_database.import_tweets(session, tweets)`
Import tweets :param session: :param tweets: :return:

31.1.11 pyTweetBot.list_actions module

`pyTweetBot.list_actions.list_actions(action_scheduler, action_type=)`
List actions :param action_scheduler: Action Scheduler object :param action_type: Filter action type

31.1.12 pyTweetBot.model_testing module

`pyTweetBot.model_testing.model_testing(data_set_file, model_file, text_size=2000, threshold=0.5)`
Test a classifier :param data_set_file: Path to the dataset file :param model_file: Path to model file if needed
:param text_size: Minimum text size :param threshold: Probability threshold

31.1.13 pyTweetBot.model_training module

`pyTweetBot.model_training.model_training(data_set_file, model_file=, model_type='NaiveBayes')`
Train a classifier on a dataset. :param data_set_file: Path to the dataset file :param model_file: Path to model file if needed :param model_type: Model's type (stat, tfidf, stat2, textblob)

31.1.14 pyTweetBot.retweet_dataset module

`pyTweetBot.retweet_dataset.retweet_dataset(config, dataset_file, search=, info=False, source='tweets')`
Get retweet data :param config: :param dataset_file: :param n_pages: :param search: Search term :param info: :return:

31.1.15 pyTweetBot.statistics_generator module

`pyTweetBot.statistics_generator.statistics_generator(twitter_connector, stats_file, n_pages, stream, info)`
Statistics generator

31.1.16 pyTweetBot.tweet_dataset module

`pyTweetBot.tweet_dataset.tweet_dataset` (*config*, *dataset_file*, *n_pages*, *info*, *rss*)

Create a tweet dataset :param config: :param tweet_connector: :return:

31.1.17 pyTweetBot.tweet_training module

`pyTweetBot.tweet_training.clean_html_text` (*to_clean*)

Clean HTML text :param to_clean: :return:

`pyTweetBot.tweet_training.tweet_training` (*dataset_file*, *model_file*=", *test*=False,
param='dp', *type*='stat')

Train a classifier on a dataset. :param config: pyTweetBot configuration object :param dataset_file: Path to the dataset file :param model_file: Path to model file if needed :param data: Title or content :param test: Test the classification success rate :param param: Model parameter (dp, ...) :param type: Model's type (stat, tfidf, stat2, textblob)

31.1.18 pyTweetBot.unfollow_dataset module

31.1.19 pyTweetBot.update_statistics module

`pyTweetBot.update_statistics.update_statistics` (*config*)

Update the statistics in the DB :param config: :return:

31.1.20 Module contents

CHAPTER 32

Indices and tables

- `genindex`
- `modindex`
- `search`

p

`pyTweetBot`, 77
`pyTweetBot.config`, 9
`pyTweetBot.create_database`, 53
`pyTweetBot.db`, 17
`pyTweetBot.db.DBConnector`, 17
`pyTweetBot.db.obj`, 16
`pyTweetBot.db.obj.Action`, 13
`pyTweetBot.db.obj.Base`, 13
`pyTweetBot.db.obj.Follower`, 13
`pyTweetBot.db.obj.Following`, 14
`pyTweetBot.db.obj.Friend`, 14
`pyTweetBot.db.obj.ImpactStatistics`, 15
`pyTweetBot.db.obj.Model`, 15
`pyTweetBot.db.obj.ModelTokens`, 15
`pyTweetBot.db.obj.Statistic`, 16
`pyTweetBot.db.obj.Tweeted`, 16
`pyTweetBot.direct_messages`, 55
`pyTweetBot.directmessages`, 19
`pyTweetBot.directmessages.directmessages`, 19
`pyTweetBot.directmessages.pyTweetBotDirectMessageAction`, 19
`pyTweetBot.directmessages.pyTweetBotDirectMessageParser`, 19
`pyTweetBot.execute_actions`, 57
`pyTweetBot.executor`, 22
`pyTweetBot.executor.ActionScheduler`, 21
`pyTweetBot.executor.ExecutorThread`, 21
`pyTweetBot.export_database`, 59
`pyTweetBot.find_follows`, 61
`pyTweetBot.find_github_tweets`, 63
`pyTweetBot.find_retweets`, 65
`pyTweetBot.find_tweets`, 67
`pyTweetBot.find_unfollows`, 69
`pyTweetBot.follower_dataset`, 75
`pyTweetBot.friends`, 23
`pyTweetBot.friends.FriendsManager`, 23
`pyTweetBot.import_database`, 76
`pyTweetBot.learning`, 28
`pyTweetBot.learning.CensorModel`, 27
`pyTweetBot.learning.Dataset`, 27
`pyTweetBot.list_actions`, 76
`pyTweetBot.mail`, 32
`pyTweetBot.mail.MailBuilder`, 31
`pyTweetBot.mail.MailSender`, 31
`pyTweetBot.model_testing`, 76
`pyTweetBot.model_training`, 76
`pyTweetBot.news`, 33
`pyTweetBot.news.GoogleNewsClient`, 33
`pyTweetBot.news.NewsParser`, 33
`pyTweetBot.patterns`, 35
`pyTweetBot.patterns.singleton`, 35
`pyTweetBot.retweet`, 37
`pyTweetBot.retweet.RetweetFinder`, 37
`pyTweetBot.retweet_dataset`, 76
`pyTweetBot.statistics_generator`, 76
`pyTweetBot.stats`, 40
`pyTweetBot.stats.TweetStatistics`, 39
`pyTweetBot.stats.UserStatistics`, 40
`pyTweetBot.templates`, 41
`pyTweetBot.tools`, 44
`pyTweetBot.tools.PageParser`, 43
`pyTweetBot.tools.strings`, 44
`pyTweetBot.tweet`, 47
`pyTweetBot.tweet.GoogleNewsHunter`, 45
`pyTweetBot.tweet.Hunter`, 45
`pyTweetBot.tweet.RSSHunter`, 45
`pyTweetBot.tweet.Tweet`, 46
`pyTweetBot.tweet.TweetFactory`, 46
`pyTweetBot.tweet.TweetFinder`, 46
`pyTweetBot.tweet.TweetPreparator`, 47
`pyTweetBot.tweet.TwitterHunter`, 47
`pyTweetBot.tweet_dataset`, 77
`pyTweetBot.tweet_training`, 77
`pyTweetBot.twitter`, 49
`pyTweetBot.twitter.TweetBotConnect`, 49
`pyTweetBot.unfollow_dataset`, 77
`pyTweetBot.update_statistics`, 77

A

Action (class in pyTweetBot.db.obj.Action), 13
 action_date (pyTweetBot.db.obj.Action.Action attribute), 13
 action_id (pyTweetBot.db.obj.Action.Action attribute), 13
 action_order (pyTweetBot.db.obj.Action.Action attribute), 13
 action_tweet_id (pyTweetBot.db.obj.Action.Action attribute), 13
 action_tweet_text (pyTweetBot.db.obj.Action.Action attribute), 13
 action_type (pyTweetBot.db.obj.Action.Action attribute), 13
 ActionAlreadyDone, 23
 ActionAlreadyExists, 21
 ActionReservoirFullError, 21
 add() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39
 add() (pyTweetBot.tweet.TweetFinder.TweetFinder method), 46
 add_follow_action() (in module pyTweetBot.find_follows), 61, 72
 add_negative() (pyTweetBot.learning.Dataset method), 28
 add_negative() (pyTweetBot.learning.Dataset.Dataset method), 27
 add_positive() (pyTweetBot.learning.Dataset method), 28
 add_positive() (pyTweetBot.learning.Dataset.Dataset method), 27
 add_tweet() (in module pyTweetBot.find_github_tweets), 63, 72
 already_tweeted() (pyTweetBot.tweet.Tweet.Tweet method), 46

B

BotConfig (class in pyTweetBot.config), 11

C

CensorModel (class in pyTweetBot.learning), 28
 CensorModel (class in pyTweetBot.learning.CensorModel), 27
 clean_html_text() (in module pyTweetBot.tweet_training), 77
 compute_tweet() (in module pyTweetBot.find_github_tweets), 63, 73
 count() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39
 create_database() (in module pyTweetBot.create_database), 53
 create_tweet_text() (in module pyTweetBot.find_github_tweets), 63, 73
 create_tweet_text_create() (in module pyTweetBot.find_github_tweets), 64, 73

D

data (pyTweetBot.learning.Dataset attribute), 28
 data (pyTweetBot.learning.Dataset.Dataset attribute), 27
 database (pyTweetBot.config.BotConfig attribute), 11
 Dataset (class in pyTweetBot.learning), 28
 Dataset (class in pyTweetBot.learning.Dataset), 27
 direct_message (pyTweetBot.config.BotConfig attribute), 11
 direct_messages() (in module pyTweetBot.direct_messages), 55

E

email (pyTweetBot.config.BotConfig attribute), 11
 execute() (pyTweetBot.db.obj.Action.Action method), 13
 execute_actions() (in module pyTweetBot.execute_actions), 57, 71
 ExecutorThread (class in pyTweetBot.executor.ExecutorThread), 21
 exists() (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic static method), 15
 exists() (pyTweetBot.db.obj.Model.Model static method), 15

exists() (pyTweetBot.db.obj.Tweeted.Tweeted static method), 16
 expect() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39
 expect_norm() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39
 export_database() (in module pyTweetBot.export_database), 59, 72

F

find_follows() (in module pyTweetBot.find_follows), 61, 72
 find_github_tweets() (in module pyTweetBot.find_github_tweets), 64, 73
 find_retweets() (in module pyTweetBot.find_retweets), 65, 74
 find_tweets() (in module pyTweetBot.find_tweets), 67, 74
 find_unfollows() (in module pyTweetBot.find_unfollows), 69, 75
 Follower (class in pyTweetBot.db.obj.Follower), 13
 follower (pyTweetBot.db.obj.Friend.Friend attribute), 14
 follower_dataset() (in module pyTweetBot.follower_dataset), 75
 follower_followed_date (pyTweetBot.db.obj.Following.Following attribute), 14
 follower_friend (pyTweetBot.db.obj.Follower.Follower attribute), 13
 follower_id (pyTweetBot.db.obj.Follower.Follower attribute), 13
 follower_last_update (pyTweetBot.db.obj.Follower.Follower attribute), 14
 Following (class in pyTweetBot.db.obj.Following), 14
 following (pyTweetBot.db.obj.Friend.Friend attribute), 14
 following_friend (pyTweetBot.db.obj.Following.Following attribute), 14
 following_id (pyTweetBot.db.obj.Following.Following attribute), 14
 following_last_update (pyTweetBot.db.obj.Following.Following attribute), 14
 forbidden_words (pyTweetBot.config.BotConfig attribute), 11
 Friend (class in pyTweetBot.db.obj.Friend), 14
 friend (pyTweetBot.db.obj.Follower.Follower attribute), 14
 friend (pyTweetBot.db.obj.Following.Following attribute), 14
 friend_contacted (pyTweetBot.db.obj.Friend.Friend attribute), 14

friend_description (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_follower (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_follower_date (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_followers_count (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_following (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_following_date (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_friends_count (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_id (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_last_update (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_location (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_screen_name (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_special (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friend_statuses_count (pyTweetBot.db.obj.Friend.Friend attribute), 14
 friends (pyTweetBot.config.BotConfig attribute), 11
 from_address() (pyTweetBot.mail.MailSender.MailSender method), 31

G

get_by_name() (pyTweetBot.db.obj.Model.Model static method), 15
 get_current_interval() (pyTweetBot.config.BotConfig method), 11
 get_friend() (pyTweetBot.db.obj.Friend.Friend static method), 14
 get_hashtags() (pyTweetBot.tweet.TwitterHunter.TwitterHunter method), 47
 get_length() (pyTweetBot.tweet.Tweet.Tweet method), 46
 get_news() (pyTweetBot.news.GoogleNewsClient.GoogleNewsClient method), 33
 get_news() (pyTweetBot.news.NewsParser.NewsParser method), 33
 get_page_title() (pyTweetBot.news.GoogleNewsClient.GoogleNewsClient method), 33
 get_random_interval() (pyTweetBot.config.BotConfig method), 11
 get_stream() (pyTweetBot.tweet.RSSHunter.RSSHunter method), 45
 get_text() (pyTweetBot.tweet.Tweet.Tweet method), 46

get_texts() (pyTweetBot.learning.Dataset method), 28
 get_texts() (pyTweetBot.learning.Dataset.Dataset method), 28
 get_tokens() (pyTweetBot.db.obj.ModelTokens.ModelToken static method), 15
 get_tweet() (pyTweetBot.tweet.Tweet.Tweet method), 46
 get_url() (pyTweetBot.tweet.Tweet.Tweet method), 46
 github (pyTweetBot.config.BotConfig attribute), 11
 google_news (pyTweetBot.config.BotConfig attribute), 11
 GoogleNewsClient (class in pyTweetBot.news.GoogleNewsClient), 33
 GoogleNewsHunter (class in pyTweetBot.tweet.GoogleNewsHunter), 45

H

handle_starttag() (pyTweetBot.news.NewsParser.NewsParser method), 33
 hashtags (pyTweetBot.config.BotConfig attribute), 11
 html (pyTweetBot.tools.PageParser.PageParser attribute), 43
 Hunter (class in pyTweetBot.tweet.Hunter), 45

I

impact_statistic_count (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic attribute), 15
 impact_statistic_hour (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic attribute), 15
 impact_statistic_id (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic attribute), 15
 impact_statistic_week_day (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic attribute), 15
 ImpactStatistic (class in pyTweetBot.db.obj.ImpactStatistics), 15
 import_actions() (in module pyTweetBot.import_database), 76
 import_database() (in module pyTweetBot.import_database), 76
 import_friends() (in module pyTweetBot.import_database), 76
 import_statistics() (in module pyTweetBot.import_database), 76
 import_tweets() (in module pyTweetBot.import_database), 76
 insert_retweet() (pyTweetBot.db.obj.Tweeted.Tweeted static method), 16
 insert_tweet() (pyTweetBot.db.obj.Tweeted.Tweeted static method), 16

is_available() (pyTweetBot.config.BotConfig method), 12
 is_awake() (pyTweetBot.config.BotConfig method), 12
 is_in() (pyTweetBot.learning.Dataset method), 28
 is_in() (pyTweetBot.learning.Dataset.Dataset method), 28

L

list_actions() (in module pyTweetBot.list_actions), 76
 load() (pyTweetBot.config.BotConfig static method), 12
 load() (pyTweetBot.learning.Dataset static method), 29
 load() (pyTweetBot.learning.Dataset.Dataset static method), 28
 load() (pyTweetBot.stats.TweetStatistics.TweetStatistics static method), 39
 load_censor() (pyTweetBot.learning.CensorModel static method), 28
 load_censor() (pyTweetBot.learning.CensorModel.CensorModel static method), 27

M

MailBuilder (class in pyTweetBot.mail.MailBuilder), 31
 MailSender (class in pyTweetBot.mail.MailSender), 31
 MAX_LENGTH (pyTweetBot.tweet.Tweet.Tweet attribute), 46
 message() (pyTweetBot.mail.MailBuilder.MailBuilder method), 31
 Model (class in pyTweetBot.db.obj.Model), 15
 model (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 15
 model_id (pyTweetBot.db.obj.Model.Model attribute), 15
 model_last_update (pyTweetBot.db.obj.Model.Model attribute), 15
 model_n_classes (pyTweetBot.db.obj.Model.Model attribute), 15
 model_name (pyTweetBot.db.obj.Model.Model attribute), 15
 model_testing() (in module pyTweetBot.model_testing), 76
 model_training() (in module pyTweetBot.model_training), 76
 ModelToken (class in pyTweetBot.db.obj.ModelTokens), 15

N

NewsParser (class in pyTweetBot.news.NewsParser), 33
 next() (pyTweetBot.learning.Dataset method), 29
 next() (pyTweetBot.learning.Dataset.Dataset method), 28
 next() (pyTweetBot.retweet.RetweetFinder.RetweetFinder method), 37
 next() (pyTweetBot.tweet.GoogleNewsHunter.GoogleNewsHunter method), 45
 next() (pyTweetBot.tweet.Hunter.Hunter method), 45
 next() (pyTweetBot.tweet.RSSHunter.RSSHunter method), 46

next() (pyTweetBot.tweet.TweetFinder.TweetFinder method), 46

next() (pyTweetBot.tweet.TwitterHunter.TwitterHunter method), 47

next_source() (pyTweetBot.tweet.TweetFinder.TweetFinder method), 46

NoFactory, 21

P

PageParser (class in pyTweetBot.tools.PageParser), 43

PageParserRetrievalError, 43

prepare_project_name() (in module pyTweetBot.find_github_tweets), 64, 74

pyTweetBot (module), 77

pyTweetBot.config (module), 9

pyTweetBot.create_database (module), 53

pyTweetBot.db (module), 17

pyTweetBot.db.DBCConnector (module), 17

pyTweetBot.db.obj (module), 16

pyTweetBot.db.obj.Action (module), 13

pyTweetBot.db.obj.Base (module), 13

pyTweetBot.db.obj.Follower (module), 13

pyTweetBot.db.obj.Following (module), 14

pyTweetBot.db.obj.Friend (module), 14

pyTweetBot.db.obj.ImpactStatistics (module), 15

pyTweetBot.db.obj.Model (module), 15

pyTweetBot.db.obj.ModelTokens (module), 15

pyTweetBot.db.obj.Statistic (module), 16

pyTweetBot.db.obj.Tweeted (module), 16

pyTweetBot.direct_messages (module), 55

pyTweetBot.directmessages (module), 19

pyTweetBot.directmessages.directmessages (module), 19

pyTweetBot.directmessages.pyTweetBotDirectMessageAction (module), 19

pyTweetBot.directmessages.pyTweetBotDirectMessenger (module), 19

pyTweetBot.execute_actions (module), 57, 71

pyTweetBot.executor (module), 22

pyTweetBot.executor.ActionScheduler (module), 21

pyTweetBot.executor.ExecutorThread (module), 21

pyTweetBot.export_database (module), 59, 72

pyTweetBot.find_follows (module), 61, 72

pyTweetBot.find_github_tweets (module), 63, 72

pyTweetBot.find_retweets (module), 65, 74

pyTweetBot.find_tweets (module), 67, 74

pyTweetBot.find_unfollows (module), 69, 75

pyTweetBot.follower_dataset (module), 75

pyTweetBot.friends (module), 23

pyTweetBot.friends.FriendsManager (module), 23

pyTweetBot.import_database (module), 76

pyTweetBot.learning (module), 28

pyTweetBot.learning.CensorModel (module), 27

pyTweetBot.learning.Dataset (module), 27

pyTweetBot.list_actions (module), 76

pyTweetBot.mail (module), 32

pyTweetBot.mail.MailBuilder (module), 31

pyTweetBot.mail.MailSender (module), 31

pyTweetBot.model_testing (module), 76

pyTweetBot.model_training (module), 76

pyTweetBot.news (module), 33

pyTweetBot.news.GoogleNewsClient (module), 33

pyTweetBot.news.NewsParser (module), 33

pyTweetBot.patterns (module), 35

pyTweetBot.patterns.singleton (module), 35

pyTweetBot.retweet (module), 37

pyTweetBot.retweet.RetweetFinder (module), 37

pyTweetBot.retweet_dataset (module), 76

pyTweetBot.statistics_generator (module), 76

pyTweetBot.stats (module), 40

pyTweetBot.stats.TweetStatistics (module), 39

pyTweetBot.stats.UserStatistics (module), 40

pyTweetBot.templates (module), 41

pyTweetBot.tools (module), 44

pyTweetBot.tools.PageParser (module), 43

pyTweetBot.tools.strings (module), 44

pyTweetBot.tweet (module), 47

pyTweetBot.tweet.GoogleNewsHunter (module), 45

pyTweetBot.tweet.Hunter (module), 45

pyTweetBot.tweet.RSSHunter (module), 45

pyTweetBot.tweet.Tweet (module), 46

pyTweetBot.tweet.TweetFactory (module), 46

pyTweetBot.tweet.TweetFinder (module), 46

pyTweetBot.tweet.TweetPreparator (module), 47

pyTweetBot.tweet.TwitterHunter (module), 47

pyTweetBot.tweet_dataset (module), 77

pyTweetBot.tweet_training (module), 77

pyTweetBot.twitter (module), 49

pyTweetBot.twitter.TweetBotConnect (module), 49

pyTweetBot.unfollow_dataset (module), 77

pyTweetBot.update_statistics (module), 77

pyTweetBotDirectmessenger (class in pyTweetBot.directmessages.pyTweetBotDirectMessenger), 19

R

raw_title (pyTweetBot.tools.PageParser.PageParser attribute), 43

reload() (pyTweetBot.tools.PageParser.PageParser method), 43

remove() (pyTweetBot.tweet.TweetFinder.TweetFinder method), 46

RequestLimitReached, 49

retweet (pyTweetBot.config.BotConfig attribute), 12

retweet_dataset() (in module pyTweetBot.retweet_dataset), 76

RetweetFinder (class in pyTweetBot.retweet.RetweetFinder), 37

rss (pyTweetBot.config.BotConfig attribute), 12

RSSHunter (class in pyTweetBot.tweet.RSSHunter), 45

run() (pyTweetBot.executor.ExecutorThread.ExecutorThread method), 21

S

save() (pyTweetBot.learning.Dataset method), 29

save() (pyTweetBot.learning.Dataset.Dataset method), 28

save() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39

scheduler (pyTweetBot.config.BotConfig attribute), 12

send() (pyTweetBot.mail.MailSender.MailSender method), 31

sendDirectMessage() (in module pyTweetBot.directmessages.directmessages), 19

set_factory() (pyTweetBot.tweet.TweetFinder.TweetFinder method), 46

set_text() (pyTweetBot.tweet.Tweet.Tweet method), 46

set_url() (pyTweetBot.tweet.Tweet.Tweet method), 46

singleton() (in module pyTweetBot.patterns.singleton), 35

start() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39

Statistic (class in pyTweetBot.db.obj.Statistic), 16

statistic_date (pyTweetBot.db.obj.Statistic.Statistic attribute), 16

statistic_followers_count (pyTweetBot.db.obj.Statistic.Statistic attribute), 16

statistic_friends_count (pyTweetBot.db.obj.Statistic.Statistic attribute), 16

statistic_id (pyTweetBot.db.obj.Statistic.Statistic attribute), 16

statistic_statuses_count (pyTweetBot.db.obj.Statistic.Statistic attribute), 16

statistics_generator() (in module pyTweetBot.statistics_generator), 76

stop() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 39

subject() (pyTweetBot.mail.MailSender.MailSender method), 31

T

targets (pyTweetBot.learning.Dataset attribute), 29

targets (pyTweetBot.learning.Dataset.Dataset attribute), 28

text (pyTweetBot.tools.PageParser.PageParser attribute), 43

title (pyTweetBot.tools.PageParser.PageParser attribute), 43

to_addresses() (pyTweetBot.mail.MailSender.MailSender method), 31

to_json() (pyTweetBot.learning.Dataset method), 29

to_json() (pyTweetBot.learning.Dataset.Dataset method), 28

token_class (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 15

token_count (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 15

token_id (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 16

token_model (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 16

token_text (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 16

token_total (pyTweetBot.db.obj.ModelTokens.ModelToken attribute), 16

Tweet (class in pyTweetBot.tweet.Tweet), 46

tweet (pyTweetBot.config.BotConfig attribute), 12

tweet_dataset() (in module pyTweetBot.tweet_dataset), 77

tweet_date (pyTweetBot.db.obj.Tweeted.Tweeted attribute), 16

tweet_id (pyTweetBot.db.obj.Tweeted.Tweeted attribute), 16

tweet_training() (in module pyTweetBot.tweet_training), 77

tweet_tweet_id (pyTweetBot.db.obj.Tweeted.Tweeted attribute), 16

tweet_tweet_text (pyTweetBot.db.obj.Tweeted.Tweeted attribute), 16

TweetAlreadyCountedException, 39

Tweeted (class in pyTweetBot.db.obj.Tweeted), 16

TweetFinder (class in pyTweetBot.tweet.TweetFinder), 46

TweetPreparator (class in pyTweetBot.tweet.TweetPreparator), 47

TweetStatistics (class in pyTweetBot.stats.TweetStatistics), 39

twitter (pyTweetBot.config.BotConfig attribute), 12

TwitterHunter (class in pyTweetBot.tweet.TwitterHunter), 47

U

UnknownEncoding, 43

update() (pyTweetBot.db.obj.ImpactStatistics.ImpactStatistic static method), 15

update_statistics() (in module pyTweetBot.update_statistics), 77

updateFollowers() (in module pyTweetBot.directmessages.directmessages), 19

url (pyTweetBot.tools.PageParser.PageParser attribute), 43

V

value() (pyTweetBot.stats.TweetStatistics.TweetStatistics method), 40

W

`wait_next_action()` (`pyTweetBot.config.BotConfig`
method), [12](#)