

# Package ‘tfse’

July 1, 2016

**Type** Package

**Title** twitter follows & selective exposure

**Version** 1.0.0

**Author** Michael W. Kearney

**Maintainer** Michael W. Kearney <mkearney@ku.edu>

**Description** this package contains R code used in my dissertation research most of the functions are designed to interact with twitter's API. this repository will one day be organized into more useful replication materials. until then, it's is a work in progress.

**License** MIT + file LICENSE

**LazyData** TRUE

**RoxygenNote** 5.0.1

**Imports** httr, jsonlite, dplyr

## R topics documented:

c.string . . . . .	2
check_rate_limit . . . . .	2
data_frame_lookup . . . . .	3
data_frame_status . . . . .	4
fromJS . . . . .	4
get_api . . . . .	4
get_followers . . . . .	5
get_followerslist . . . . .	5
get_followers_max . . . . .	6
get_friends . . . . .	6
get_friendships . . . . .	7
get_friendslist . . . . .	7
get_friends_max . . . . .	8
get_friends_ply . . . . .	9
get_list . . . . .	9
get_lookup . . . . .	10
get_lookup_max . . . . .	10
get_package_pdf . . . . .	11
get_retweets . . . . .	11
get_timeline . . . . .	12
get_token . . . . .	13
get_wave . . . . .	14

get_wave_data . . . . .	14
gg_cols . . . . .	15
is_screen_name . . . . .	15
load_tokens . . . . .	15
prep_vector . . . . .	16
search_tweets . . . . .	16
sn2id . . . . .	17
top_tweet_words . . . . .	18
try_catch . . . . .	18
TWIT . . . . .	19
wave_path_name . . . . .	19
which_ids . . . . .	20
<b>Index</b>	<b>21</b>

---

c.string	<i>c.string</i>
----------	-----------------

---

**Description**

c.string

**Usage**

```
## S3 method for class 'string'  
c(x)
```

**Arguments**

x                      Character vector

**Value**

Printed output via cat() function for copy/paste of long strings

---

check_rate_limit	<i>check_rate_limit</i>
------------------	-------------------------

---

**Description**

check\_rate\_limit

**Usage**

```
check_rate_limit(type, token, seconds = FALSE)
```

**Arguments**

type	If not missing, which returns entire rate limit request object, type returns specific values; e.g., type = "lookup" returns remaining limit for user lookup requests; type = "followers" returns remaining limit for follower id requests; type = "friends" returns remaining limit for friend id requests.
token	An OAuth token (1.0 or 2.0)
seconds	logical, indicating whether to return unix seconds

**Value**

response Rate limit response object or specific value of remaining requests

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

data_frame_lookup	<i>data_frame_lookup</i>
-------------------	--------------------------

---

**Description**

data\_frame\_lookup

**Usage**

```
data_frame_lookup(x)
```

**Arguments**

x	json resposne object from user lookup Twitter API call.
---	---

**Value**

data frame

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

data_frame_status	<i>data_frame_status</i>
-------------------	--------------------------

---

**Description**

data\_frame\_status

**Usage**

data\_frame\_status(x)

**Arguments**

x	json object from search tweets
---	--------------------------------

---

fromJS	<i>fromJS</i>
--------	---------------

---

**Description**

fromJS

**Usage**

fromJS(x)

**Arguments**

x	json object
---	-------------

---

get_api	<i>get_api</i>
---------	----------------

---

**Description**

get\_api

**Usage**

get\_api(url, token = NULL)

**Arguments**

url	API url address.
-----	------------------

**Value**

Response formatted as nested list. Assumes response object is json object.

---

get_followers	<i>get_followers</i>
---------------	----------------------

---

**Description**

get\_followers

**Usage**

```
get_followers(user, token, page = "-1", stringify = TRUE)
```

**Arguments**

user	Screen name or user id of target user.
token	OAuth token (1.0 or 2.0)
page	Default page = -1 specifies first page of json results. Other pages specified via cursor values supplied by Twitter API response object.
stringify	logical, indicating whether to return user ids as strings (some ids are too long to be read as numeric). Defaults to TRUE

**Value**

user ids

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_followerslist	<i>get_followerslist</i>
-------------------	--------------------------

---

**Description**

Returns a cursored collection of user objects for users following the specified user.

**Usage**

```
get_followerslist(user, token, page = "-1")
```

**Arguments**

user	Screen name or user id of target user.
token	OAuth token (1.0 or 2.0)
page	Default page = -1 specifies first page of json results. Other pages specified via cursor values supplied by Twitter API response object.

**Value**

json user object (nested list)

**See Also**

See <https://api.twitter.com/1.1/followers/list.json>.

---

get_followers_max	<i>get_followers_max</i>
-------------------	--------------------------

---

**Description**

get\_followers\_max

**Usage**

```
get_followers_max(user, tokens)
```

**Arguments**

user	Screen name or user id of target user
tokens	OAuth tokens (1.0 or 2.0)

**Value**

user ids

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_friends	<i>get_friends</i>
-------------	--------------------

---

**Description**

get\_friends

**Usage**

```
get_friends(user, token, page = "-1", stringify = TRUE)
```

**Arguments**

user	Screen name or user id of target user.
token	OAuth token (1.0 or 2.0)
page	Default page = -1 specifies first page of json results. Other pages specified via cursor values supplied by Twitter API response object.
stringify	logical, indicating whether to return user ids as strings (some ids are too long to be read as numeric). Defaults to TRUE

**Value**

friends User ids for everyone a user follows.

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_friendships	<i>get_friendships</i>
-----------------	------------------------

---

**Description**

Returns detailed information about the relationship between two arbitrary users.

**Usage**

```
get_friendships(source, target, token)
```

**Arguments**

source	Screen name or user id of source user.
target	Screen name or user id of target user.
token	OAuth token (1.0 or 2.0)

**Value**

json object (nested list)

**See Also**

See <https://api.twitter.com/1.1/friendships/show.json>.

---

get_friendslist	<i>get_friendslist</i>
-----------------	------------------------

---

**Description**

Returns a cursored collection of user objects for every user the specified user is following (otherwise known as their “friends”).

**Usage**

```
get_friendslist(user, token, page = "-1")
```

**Arguments**

user	Screen name or user id of target user.
token	OAuth token (1.0 or 2.0)
page	Default page = -1 specifies first page of json results. Other pages specified via cursor values supplied by Twitter API response object.

**Value**

json user object (nested list)

**See Also**

See <https://api.twitter.com/1.1/friends/list.json>.

---

get_friends_max	<i>get_friends_max</i>
-----------------	------------------------

---

**Description**

get\_friends\_max

**Usage**

```
get_friends_max(user_ids, tokens, start = 1, stringify = TRUE,
  verbose = TRUE)
```

**Arguments**

user_ids	Data frame with column name "screen_name"
tokens	OAuth tokens (1.0 or 2.0)
start	Starting value (nth user id)
stringify	logical, indicating whether to return user ids as strings
verbose	default behavior verbose = TRUE prints asterisk for every 15 (or max of one token) user networks collected. Set verbose = FALSE to run function silently. (some ids are too long to be read as numeric). Defaults to TRUE.

**Value**

friends List of user ids each user follows.

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.



---

get_friends_ply	<i>get_friends_ply</i>
-----------------	------------------------

---

**Description**

lapply() version of get\_friends\_max()

**Usage**

```
get_friends_ply(user_ids, tokens, start = 1)
```

**Arguments**

user_ids	Data frame with column name "screen_name"
tokens	OAuth tokens (1.0 or 2.0)
start	Starting value (nth user id)

**Value**

friends List of user ids each user follows.

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_list	<i>get_list</i>
----------	-----------------

---

**Description**

get\_list

**Usage**

```
get_list(slug, owner, token, count = 5000, cursor = "-1")
```

**Arguments**

slug	Name of Twitter list
owner	Screen name of list owner
token	An OAuth token (1.0 or 2.0)
count	Maximum number of users to return (cannot be higher than 5000).
cursor	Select next or previous page or results using cursor value from returned json object

**Value**

json response object as nested list

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_lookup	<i>get_lookup</i>
------------	-------------------

---

**Description**

get\_lookup

**Usage**

```
get_lookup(users, token, df = TRUE, skip = TRUE, entities = FALSE)
```

**Arguments**

users	Screen name or user id of target users.
token	OAuth tokens (1.0 or 2.0)
df	logical, indicating whether to format response as data frame

**Value**

response object

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_lookup_max	<i>get_lookup_max</i>
----------------	-----------------------

---

**Description**

get\_lookup\_max

**Usage**

```
get_lookup_max(ids, tokens, start = 1)
```

**Arguments**

ids	User id or screen name of target user.
tokens	OAuth tokens (1.0 or 2.0)
start	First (nth) id

**Value**

response object

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

get_package_pdf	<i>get_package_pdf</i>
-----------------	------------------------

---

**Description**

get\_package\_pdf

**Usage**

```
get_package_pdf(package, update = FALSE, open = TRUE)
```

**Arguments**

package	Name of target package.
update	logical, indicating whether to replace old manual
open	logical, indicating whether to automatically view preview of pdf

---

get_retweets	<i>get_retweets</i>
--------------	---------------------

---

**Description**

Returns a collection of the 100 most recent retweets of the tweet specified by the id parameter.

**Usage**

```
get_retweets(tweet_id, count = 100, trim_user = TRUE, token)
```

**Arguments**

tweet_id	required, The numerical ID of the desired status
count	optional, Specifies the number of records to retrieve. Must be less than or equal to 100.
trim_user	optional, When set to TRUE each tweet returned in a timeline will include a user object including only the status authors numerical ID. Omit this parameter to receive the complete user object.
token	OAuth token (1.0 or 2.0)

**Value**

json object

**See Also**

<https://api.twitter.com/1.1/statuses/retweets/:id.json>

---

get\_timeline

get\_timeline

---

**Description**

Returns a collection of the most recent Tweets posted by the user indicated by the `screen_name` or `user_id` parameters. User timelines belonging to protected users may only be requested when the authenticated user either “owns” the timeline or is an approved follower of the owner. The timeline returned is the equivalent of the one seen when you view a user’s profile on twitter.com. This method can only return up to 3,200 of a user’s most recent Tweets. Native retweets of other statuses by the user is included in this total, regardless of whether `include_rts` is set to false when requesting this resource.

**Usage**

```
get_timeline(user, since_id = NULL, count = NULL, max_id = NULL,
             trim_user = NULL, exclude_replies = FALSE, contributor_details = TRUE,
             include_rts = TRUE, token)
```

**Arguments**

<code>user</code>	The <code>screen_name</code> or ID of the user for whom to return results for.
<code>since_id</code>	Returns results with an ID greater than (that is, more recent than) the specified ID. There are limits to the number of Tweets which can be accessed through the API. If the limit of Tweets has occurred since the <code>since_id</code> , the <code>since_id</code> will be forced to the oldest ID available.
<code>count</code>	Specifies the number of tweets to try and retrieve, up to a maximum of 200 per distinct request. The value of <code>count</code> is best thought of as a limit to the number of tweets to return because suspended or deleted content is removed after the count has been applied. We include retweets in the count, even if <code>include_rts</code> is not supplied. It is recommended you always send <code>include_rts=1</code> when using this API method.
<code>max_id</code>	Returns results with an ID less than (that is, older than) or equal to the specified ID.
<code>trim_user</code>	When set to either true, t or 1, each tweet returned in a timeline will include a user object including only the status authors numerical ID. Omit this parameter to receive the complete user object.
<code>exclude_replies</code>	This parameter will prevent replies from appearing in the returned timeline. Using <code>exclude_replies</code> with the <code>count</code> parameter will mean you will receive up-to count tweets — this is because the <code>count</code> parameter retrieves that many tweets before filtering out retweets and replies. This parameter is only supported for JSON and XML responses.
<code>include_rts</code>	When set to false, the timeline will strip any native retweets (though they will still count toward both the maximal length of the timeline and the slice selected by the <code>count</code> parameter). Note: If you’re using the <code>trim_user</code> parameter in conjunction with <code>include_rts</code> , the retweets will still contain a full user object.

token	OAuth token (1.0 or 2.0)
contributor_details	This parameter enhances the contributors element of the status response to include the screen_name of the contributor. By default only the user_id of the contributor is included.

**Value**

json object (nested list)

**See Also**

See [https://api.twitter.com/1.1/statuses/user\\_timeline.json](https://api.twitter.com/1.1/statuses/user_timeline.json).

---

get_token	<i>get_token</i>
-----------	------------------

---

**Description**

get\_token

**Usage**

get\_token(app, consumer\_key, consumer\_secret)

**Arguments**

app	Name of user created Twitter application
consumer_key	Application API key
consumer_secret	Application API secret User-owned app must have Read and write access level and Callback URL of http://127.0.0.1:1410.

**Value**

twitter oauth 1.0 token

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

`get_wave`*get\_wave*

---

**Description**`get_wave`**Usage**`get_wave(ids, tokens)`**Arguments**

<code>ids</code>	Vector of user ids
------------------	--------------------

**Value**

friends List of user ids each user follows.

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

`get_wave_data`*get\_wave\_data*

---

**Description**`get_wave_data`**Usage**`get_wave_data(wave, ids, tokens)`**Arguments**

<code>wave</code>	Wave number
-------------------	-------------

**Value**

data Saved wave data and updated running data set

---

gg_cols	<i>gg_cols</i>
---------	----------------

---

**Description**

gg\_cols

**Usage**

gg\_cols(n)

**Arguments**

n                      number of colors to be generated

**Value**

colors a la ggplot

---

---

is_screen_name	<i>is_screen_name</i>
----------------	-----------------------

---

**Description**

is\_screen\_name

**Usage**

is\_screen\_name(x)

**Arguments**

x                      Twitter user id or screen name

**Value**

logical value indicating whether object is screen name [or user ID]

---

---

load_tokens	<i>load_tokens</i>
-------------	--------------------

---

**Description**

load\_tokens

**Usage**

load\_tokens()

**Value**

twitter oauth 1.0 tokens

---

prep_vector	<i>prep_vector</i>
-------------	--------------------

---

**Description**

prep\_vector

**Usage**

```
prep_vector(x)
```

**Arguments**

x	data to be vectorized
---	-----------------------

---

search_tweets	<i>search_tweets</i>
---------------	----------------------

---

### Description

Returns a collection of relevant Tweets matching a specified query. Please note that Twitter’s search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface. In API v1.1, the response format of the Search API has been improved to return Tweet objects more similar to the objects you’ll find across the REST API and platform. However, perspectival attributes (fields that pertain to the perspective of the authenticating user) are not currently supported on this endpoint. To learn how to use Twitter Search effectively, consult our guide to Using the Twitter Search API. See Working with Timelines to learn best practices for navigating results by since\_id and max\_id.

### Usage

```
search_tweets(q, geocode = NULL, lang = NULL, locale = NULL,
  result_type = "recent", count = 100, until = NULL, since_id = NULL,
  max_id = NULL, include_entities = TRUE, token)
```

### Arguments

q	required, A UTF-8, URL-encoded search query of 500 characters maximum, including operators. Queries may additionally be limited by complexity.
geocode	optional, Returns tweets by users located within a given radius of the given latitude/longitude. The location is preferentially taking from the Geotagging API, but will fall back to their Twitter profile. The parameter value is specified by “latitude,longitude,radius”, where radius units must be specified as either “mi” (miles) or “km” (kilometers). Note that you cannot use the near operator via the API to geocode arbitrary locations; however you can use this geocode parameter to search near geocodes directly. A maximum of 1,000 distinct “sub-regions” will be considered when using the radius modifier.
lang	optional, Restricts tweets to the given language, given by an ISO 639-1 code. Language detection is best-effort.



locale	optional, Specify the language of the query you are sending (only ja is currently effective). This is intended for language-specific consumers and the default should work in the majority of cases.
result_type	optional, Specifies what type of search results you would prefer to receive. The current default is "mixed." Valid values include "mixed" to include both popular and real time results in the response, "recent" to return only the most recent results in the response, and "popular" to return only the most popular results in the response.
count	optional, The number of tweets to return per page, up to a maximum of 100. Defaults to 15. This was formerly the "rpp" parameter in the old Search API.
until	optional, Returns tweets created before the given date. Date should be formatted as YYYY-MM-DD. Keep in mind that the search index has a 7-day limit. In other words, no tweets will be found for a date older than one week. Example Values: "2015-07-19".
since_id	optional, Returns results with an ID greater than (that is, more recent than) the specified ID. There are limits to the number of Tweets which can be accessed through the API. If the limit of Tweets has occurred since the since_id, the since_id will be forced to the oldest ID available.
max_id	optional, Returns results with an ID less than (that is, older than) or equal to the specified ID.
include_entities	optional, The entities node will be disincluded when set to false.
token	OAuth token (1.0 or 2.0)
callback	optional, If supplied, the response will use the JSONP format with a callback of the given name. The usefulness of this parameter is somewhat diminished by the requirement of authentication for requests to this endpoint. Example Values: "processTweets"

**Value**

json object

**See Also**

<https://api.twitter.com/1.1/search/tweets.json>

---

 sn2id

*sn2id*


---

**Description**

sn2id

**Usage**

sn2id(screen\_name)

**Arguments**

screen\_name      Twitter handle

**Value**

response Twitter account user id

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

top_tweet_words	<i>top_tweet_words</i>
-----------------	------------------------

---

**Description**

top\_tweet\_words

**Usage**

```
top_tweet_words(tweets_text, min = 3, exclude_words = NULL)
```

**Arguments**

tweets_text	character vector of tweets text
min	minimum number of occurrences to include in returned object
exclude_words	other words to exclude

**Value**

list object with top mentions and top words

---

try_catch	<i>try_catch</i>
-----------	------------------

---

**Description**

try\_catch

**Usage**

```
try_catch(x)
```

**Arguments**

x	function call
---	---------------

---

TWIT	<i>TWIT</i>
------	-------------

---

**Description**

TWIT

**Usage**

```
TWIT(query, parameters = NULL, token, parse = TRUE, version = "1.1")
```

**Arguments**

query	Twitter API request type string. e.g, "friends/ids" calls Twitter API to return information about a user's friend network (i.e., accounts followed by a user).
parameters	Additional parameters passed along to API call
token	An OAuth token (1.0 or 2.0)

**Value**

json response object as nested list

**See Also**

See <https://dev.twitter.com/overview/documentation> for more information on using Twitter's API.

---

wave_path_name	<i>wave_path_name</i>
----------------	-----------------------

---

**Description**

wave\_path\_name

**Usage**

```
wave_path_name(data_folder = "/Users/mwk/r/tfse/study/data/")
```

**Value**

Printed output via cat() function for copy/paste of long strings

---

which_ids	<i>which_ids</i>
-----------	------------------

---

**Description**

Returns integer values. Used for get\_friends function.

**Usage**

```
which_ids(n, max_users = 3000, token = NULL)
```

**Arguments**

n	starting number for users
max_users	max number of user ids (if rate limit exceeds remaining number of users, this sets upper ceiling and reduces likelihood of API request errors)
token	Specify token if there is reason to believe current remaning friend list request is below the rate limit max of 15. This rate limit resets every 15 minutes, so this is usually not necessary. Checking rate limits does not reduce the number of available requests, but it does slow things down.

**Value**

integers used to identify 15 (or token max given rate limits) users from provided list of user ids

# Index

c.string, [2](#)  
check\_rate\_limit, [2](#)  
  
data\_frame\_lookup, [3](#)  
data\_frame\_status, [4](#)  
  
fromJS, [4](#)  
  
get\_api, [4](#)  
get\_followers, [5](#)  
get\_followers\_max, [6](#)  
get\_followerslist, [5](#)  
get\_friends, [6](#)  
get\_friends\_max, [8](#)  
get\_friends\_ply, [9](#)  
get\_friendships, [7](#)  
get\_friendslist, [7](#)  
get\_list, [9](#)  
get\_lookup, [10](#)  
get\_lookup\_max, [10](#)  
get\_package\_pdf, [11](#)  
get\_retweets, [11](#)  
get\_timeline, [12](#)  
get\_token, [13](#)  
get\_wave, [14](#)  
get\_wave\_data, [14](#)  
gg\_cols, [15](#)  
  
is\_screen\_name, [15](#)  
  
load\_tokens, [15](#)  
  
prep\_vector, [16](#)  
  
search\_tweets, [16](#)  
sn2id, [17](#)  
  
top\_tweet\_words, [18](#)  
try\_catch, [18](#)  
TWIT, [19](#)  
  
wave\_path\_name, [19](#)  
which\_ids, [20](#)