twitterapiR Wraper Function for R

Nelson Tang, Ling Xiang Zou, Nyanda Redwood

1. Introduction

The intent of the ***twitterapiR*** is to provide a quick exploratory service of the resources provided by ***Twitter***, a popular social media platform. It does so by providing access to the Twitter API via R.

twitterapiR outputs a general DataFrame as well disaggregated DataFrames for the total number of followers per user as well the names of the friends of a user. Both these subsets include the user's screen name. The latter subset is capped to a maximum number of 195 friends per user and displays the user's name separately from the user's screen name.

2. Authentication

You will need to follow these instructions to continue. We are accessing ***Twitter*** programatically. ***twitterapiR*** uses the **httr** package under the hood to manage this.
The first step is to create a ***Twitter*** application for yourself. To do so, go to the Developer Platform's page and log in.

Follow the instructions and fill in some basic information. After your project is created, you can generate your consumer API and secret key.

In your R session, you will want to do the following using the appropriate values from the web page:

```
set_bearer("API key", "API secret")
```

This will authenticate via **httr**, we recommend looking at the Token man page of this package for more information regarding how to manage the authentication and caching processes.

3. Getting Started

This document is intended to describe the usage of each function and to show examples of each function. To explore the source code or report some issues, we recommend going to our GitHub. Before exploring our functionality, please make sure you complete the following steps.

```
# install package
remotes::install_github("tangaot/twitterapiR")
```

```
library(twitterapiR)
```

```
set_bearer("API key", "API secret")
```

4. Exploring *twitterapiR*

4.1 Necessary Environtment Variables

The **set_bearer** function will set key and secret key as environment variables. The key and secret key are always required. These are essentially the username and password for Twitter API. You can refer to **Authentication** to generate the key and secret key. If the key and secret key are successfully set, the function will return `TRUE`. Otherwise, it will return `FALSE`.

```
set_bearer("API key", "API secret")
```

The **get_bearer** function is a helper function that can be used to generate the bearer key by using `API key` and `API secret`. It is embedded in the functions **searchTweets**, **user_friends**, and **followersCount**. The bearer key allows the users to query information from Twitter.

```
get_bearer()
```

4.2 Exploring Friends of a User

The **user_friends** function can be used to get some of the names of the friends of a user. A friend is being understood here as an individual who reciprocates the act of following another person. The **user_friends** function takes two arguments: `screen_name` and `number`.

To get the friends of a user, run the command **user_friends**. This will only work correctly with users that have public profiles, or if you are authenticated and granted access. The **user_friends** function returns a DataFrame with the name of the user, screen name of the user, and names of the friends of the user.

```
user_friends(screen_name = "BarackObama", number = 10)
```

4.3 Exploring the Number of Followers of a User

The **followersCount** function can be used to return the total number of followers of a user. It does so by using a user's screen name as its sole argument. It returns the user's screen name and the total number of followers of a user in a DataFrame. The idea here is to cater to those curious about the number of followers a user of interest has.

To get the number of followers a user has, run the command **followersCount**. This will only work correctly with users that have public profiles, or if you are authenticated and granted access.

```
followersCount(screen_name = "BarackObama")
```

4.4 Search Tweets

The **searchTweets** function can be used to search for related tweets that match a specified string. The argument for `searchString` is a string of 100 characters, maximum, that pertains to your desired search terms in a tweet. The argument can also contain hashtags if you are interested in finding specific hashtags. The `count` will restrict the maximum number of returned tweets, and it defaults to 5. The `resultType` will specify the kind of search result that you want, and it defaults to "mixed." Valid values include:

- mixed: includes both popular and real time results in the response
- recent: returns only the most recent results in the response
- popular: returns only the most popular results in the response

The function will return a DataFrame containing the following columns:

- created_time: the time the tweet was created
- user_name: the name of who posted the tweets
- user_screen_name: the screen name of who posted the tweet

- user_followers_count: the number of followers that posted the tweet
- text: the text within the tweet
- truncated: indicates if the tweet is truncated
- favorited: indicates if the tweet is favorited
- retweeted: indicates if the tweet was retweeted
- favorite_count: the number of times the tweet was favorited
- retweet_count: the number of times the tweet was retweeted

```
searchTweets("#ubc", resultType = "mixed", count = 5)
```