Explanation Twitter Scraper

In this document, we will give you an explanation how the Twitter scraper works.

Installations and imports

Before we can use the scraper, we need some installs and imports:

```
# Installs
!pip install git+https://github.com/JustAnotherArchivist/snscrape.git
!pip install snscrape
!pip install pandas
!pip install dropbox
!pip install pathlib
!pip install regex
```

```
# Imports
import snscrape.modules.twitter as sntwitter
import pandas as pd
from time import sleep
import pathlib
import dropbox
import re
import csv
import os
```

Function

Our function looks pretty darn badass, but we will explain its simplicity to you


```
# Function for scraping

def get_twitterdata(filename,
                    season,
                    period,
                    maxTweets = 100,
                    searchq = ['#afcajax'],
                    since = '2021-01-01', until='2021-01-03'):

    # Begin with X tweets to search in order to see if it works without taking too much time
    timefilter= 'since:' + since +' until:'+until
    print("Hi! I'm starting up!")
    print('for time period: '+ timefilter)
    # Creating list to append tweet data to
    tweets_list1 = []

    # Using SNScraper to scrape data and append tweets to list

    for i,tweet in enumerate(sntwitter.TwitterSearchScraper(searchq + ' ' + timefilter).get_items()):
        if i>maxTweets: #stops when amount of tweets is at its maximum
            break

        tweets_list1.append([season, period, tweet.url, tweet.date, tweet.id, tweet.content, tweet.user.username, tweet.replyCour
    print('Done scraping! Saving now :)')

    # Creating a dataframe from the tweets list above
    df_tweets1 = pd.DataFrame(tweets_list1, columns=['Season', 'Period','URL', 'Datetime', 'Tweet Id', 'Text', 'Username', 'Repli
    df_tweets1.to_csv(filename, sep=',', index=False)

    print(f'Done with data from {season} {period}!')
```

1. This is the standard input of the function, as you can see we are going to scrape a max of 100 tweets with the hashtag #afcajax in it for a particular weekend in 2021.
2. This is how the timefilter is build up: ‘since: 2021-01-01 until: 2021-01-03’

3. This is where the sntwitter scraper is going to do its work. It combines the search (#afcajax) with the timefilter: #afcajax since:2021-01-01 until:2021-01-03. If you put this line exactly in the twitter search function, you get the tweet you are going to scrape:
    a. 

4. Here you can see which variables are scraped from the tweets, how the column names are named in the csv file and how the csv file is saved.

Running the function

Now you can run the function as many times as you want, with different variables:

```
# Season 19/20, period 1
get_twitterdata(filename= 'season19-20period1max100.csv',
                searchq= '#AdoDenHaag OR #AFCAjax OR #AZalkmaar OR #FCEmmen OR #FCGroningen OR #FCTwente OR #FCUtrecht OR #Fey
                since = '2019-11-22',
                until = '2019-11-24',
                season= 'season19/20',
                period= 'period1')

sleep(5)

# Season 19/20, period 2
get_twitterdata(filename= 'season19-20period2max100.csv',
                searchq= '#AdoDenHaag OR #AFCAjax OR #AZalkmaar OR #FCEmmen OR #FCGroningen OR #FCTwente OR #FCUtrecht OR #Fey
                since = '2020-01-24',
                until = '2020-01-26',
                season= 'season19/20',
                period= 'period2')
```

Merging

The output of the scraping are multiple csv files, if you want to merge these files into one file, you can do this here:

Now merging and exporting local

```
Find_All_CSVs = ['season19-20period1max100.csv',
                 'season19-20period2max100.csv',
                 'season20-21period1max100.csv',
                 'season20-21period2max100.csv']

#combine all files in the list

combined_csv = pd.concat([pd.read_csv(f,header=0) for f in Find_All_CSVs])

combined_csv.head()

combined_csv.to_csv("merged_tweets_eredivisiemax100.csv", quotechar='"',
          quoting=csv.QUOTE_ALL, index=False, encoding='utf-8')

print("all done! twitter data available in your directory")
```

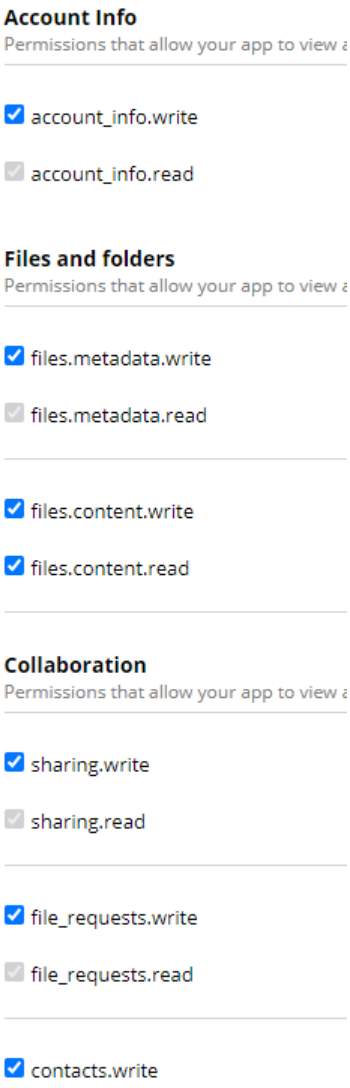
Dropbox

The last part of the code is exporting the merged file to your dropbox, so it is save in the cloud ☺

Now exporting to Dropbox

```
]: class TransferData:
    def __init__(self, access_token):
        self.access_token = access_token

    def upload_file(self, file_from, file_to):
        """upload a file to Dropbox using API v2
        """
        dbx = dropbox.Dropbox(self.access_token)

        with open(file_from, 'rb') as f:
            dbx.files_upload(f.read(), file_to)

def main():
    access_token = 'Dropbox-Token'
    transferData = TransferData(access_token)

    file_from = 'merged_tweets_eredivisie.csv'
    file_to = '/Apps/dPrep/merged_tweets_eredivisie.csv'  # The full path to upload the file to, including the file name

    # API v2
    transferData.upload_file(file_from, file_to)

if __name__ == '__main__':
    main()

print("All done, check out the dropbox folder: ")
```

1. First, you have to create a Dropbox API token (https://www.dropbox.com/developers/apps). You need to create an app, make a map inside this app, get the API access token for this app and fill it in at point 1. More information about creating a Dropbox app here: https://www.dropbox.com/developers/documentation. Youtube also is your friend ;)
    a. Under app permissions, don't forget to get everything like this before creating your access token:



b.

2. Fill in the name of the merged csv file, and the exact path of the dropbox map structure of the created app. So fill in the whole path of where you want your file to be put.