WS-AllScraping

January 7, 2019

1 Web Scraper

1.1 Versions

```
In [1]: import platform

        print('Python version: {}'.format(platform.python_version()))
```

```
Python version: 3.6.4
```

1.2 Import Libraries

```
In [42]: from lxml import html
         import requests
         import pandas as pd
         import time
```

1.3 Functions

```
In [277]: def findStars(x,site):
              if site.lower() == 'tripadvisor':
                  x2 = str(x).replace('>', ' ').split()
                  if ('bubble_5"' in x2):
                      return 0.5
                  elif ('bubble_10"' in x2):
                      return 1
                  elif ('bubble_15"' in x2):
                      return 1.5
                  elif ('bubble_20"' in x2):
                      return 2
                  elif ('bubble_25"' in x2):
                      return 2.5
                  elif ('bubble_30"' in x2):
                      return 3
                  elif ('bubble_35"' in x2):
                      return 3.5
                  elif ('bubble_40"' in x2):
```

```
            return 4
        elif ('bubble_45"' in x2):
            return 4.5
        elif ('bubble_50"' in x2):
            return 5
        else:
            return 0
    elif site.lower() == 'yelp':
        x2 = str(x)
        if ('0.5 star' in x2):
            return 0.5
        elif ('1.0 star' in x2):
            return 1
        elif ('1.5 star' in x2):
            return 1.5
        elif ('2.0 star' in x2):
            return 2
        elif ('2.5 star' in x2):
            return 2.5
        elif ('3.0 star' in x2):
            return 3
        elif ('3.5 star' in x2):
            return 3.5
        elif ('4.0 star' in x2):
            return 4
        elif ('4.5 star' in x2):
            return 4.5
        elif ('5.0 star' in x2):
            return 5
        else:
            return 0

# Diagnostics
def diagnostics(silent,*args):
    '''
    This function checks that the lists given as arguments are of equal sizes
    args: An arbitrary number of lists
    silent: A boolean indicating whether diagnostic results are to be displayed
    '''

    if not silent:
        print('Diagnostics: Checking if dataframes are of equal size...')
    [print('Size: {}'.format(len(i))) for i in args if not silent]

    l = len(args[0])

    for i in args:
        if len(i) != l:
```

```
            if not silent:
                print('Unequal Sizes!')
            return False
    if not silent:
        print('Diagnostics complete!')
    return True

def webscrape(url, site, silent):
    '''
    This functioni scrapes relevant review tags from a website url
    url: A string url
    site: A string indicating the site name to be scraped
    silent: A boolean indicating whether diagnostic results are to be displayed
    '''
    # A variable to store the success of the read
    success = False

    # Get the request object from the server
    page = requests.get(url)

    # Convert the request content to an html object
    top = html.fromstring(page.content)

    # These are the class names
    rev_class_1 = ''
    rev_class_2 = ''
    rat_class = ''
    title_class = ''
    dat_class = ''
    dat_class_2 = ''

    if site.lower() == 'tripadvisor':
        rev_class_1 = 'innerBubble'
        rev_class_2 = 'entry'
        rat_class = 'review-container'
        titl_class = 'noQuotes'
        dat_class = 'ratingDate'

        # Get all the innerBubble classes which contain the reviews as well as any r
        reviews = top.find_class(rev_class_1)

        # Loop through the reviews
        reviews_array = []

        for i in reviews:
            reviews_array.append((i.find_class(rev_class_2)[0]).text_content())

        # Get all the review containers
```

```
        ratings=top.find_class(rat_class)

        # Within each review container is a class, the name of which determines the
        # We use the findStars function to determine the rating from the class name
        ratings_array = []
        for i in ratings:
            ratings_array.append(findStars(html.tostring(i),site))

        # The titles of the reviews are within the 'noQuotes' tags
        titles=top.find_class(titl_class)

        # Get the titles
        titles_array = []
        for i in titles:
            titles_array.append(i.text_content())

        # Get the rating date tags
        dates=top.find_class(dat_class)

        # Get the dates
        dates_array=[]
        for i in dates:
            dates_array.append(i.text_content())

        # Diagnostics
        success = diagnostics(silent,ratings_array,reviews_array,dates_array,titles_a

    elif site.lower() == 'yelp':
        rev_class_1 = 'review-content'
        rev_class_2 = 'p'
        rat_class = 'biz-rating'
        dat_class_2 = 'rating-qualifier'

        # Get all the innerBubble classes which contain the reviews as well as any r
        reviews = top.find_class(rev_class_1)

        # Loop through the reviews
        reviews_array = []

        for i in reviews:
            reviews_array.append(i.find(rev_class_2).text_content())

        # Set empty the titles
        titles_array = reviews_array.copy()

        # Within each review-content is a class called biz-rating, the name of which
        # We use the findStars function to determine the rating from the class name
        ratings_array = []
```

```
            for i in [getattr(i,'find_class')(rat_class)[0] for i in reviews]:
                ratings_array.append(findStars(html.tostring(i),'yelp'))

            # Get the dates. When a review is updated, the word updated review is presen
            dates_array=[]
            for i in reviews:
                dates_array.append((i.find_class(dat_class_2)[0].text_content()).replace

            # Diagnostics
            success = diagnostics(silent,ratings_array,reviews_array,dates_array)

        else:
            print('The site {} is not supported'.format(site))
            return False

        # Convert to a dataframe
        df_review = pd.DataFrame(reviews_array, columns=['Review'])
        df_ratings = pd.DataFrame(ratings_array, columns=['Rating'])
        df_titles = pd.DataFrame(titles_array, columns=['title'])
        df_reviewdates = pd.DataFrame(dates_array, columns=['date'])

        # Consolidate into a dataframe
        df_fullreview = pd.concat([df_review,df_titles,df_ratings['Rating'],df_reviewdate
        df_fullreview.dropna(inplace=True)

        # Combine review and title into a single column
        df_fullreview['fullreview'] = df_fullreview['Review'] + ' ' + df_fullreview['titl

        return df_fullreview,success

    def fullscraper(site, first_url, url1, url2, increment_string1, increment_string2, to
        '''
        This function increments the site url to the next page according to update crite
        The full url of subsequent pages is url = url1 + increment_string1 + url2 + incr
        of error in reading information, attempts are made to re-read data.
        first_url: A string url. The main url page
        url1, url2: The static parts of the urls that do not change in incrementation
        increment_string1, increment_string2: The parts of the url that change
        total_pages: The number of total pages. Integer
        output_file: The file name to output. If empty string, it doesn't save a file
        site: A string indicating the site name to be scraped
        '''
        success = False

        # Main data frame
        df = pd.DataFrame()
```

```
    # Progress output
    print('Getting reviews ' + str(0)+'/ '+str(total_pages))

    # url incrementation differs per website
    if site.lower() in ['tripadvisor','yelp']:
        while not success:
            df,success = webscrape(first_url,site,False)
            if not success:
                print('Error in reading - Re-reading')

            # Wait for 1 second
            time.sleep(1)

        print('Getting reviews ' + str(1)+'/ '+str(total_pages))

        for i in range(1,total_pages):
            success = False
            url_temp = url1 + increment_string1 + str(i*increment) + increment_string

            if (i%10 == 0) or (i == total_pages):
                while not success:
                    df_temp,success = webscrape(url_temp,site,False)
                    if not success:
                        print('Error in reading - Re-reading')

                    # Wait for 1 second
                    time.sleep(1)
            else:
                while not success:
                    df_temp,success = webscrape(url_temp,site,True)
                    if not success:
                        print('Error in reading - Re-reading')

                    # Wait for 1 second
                    time.sleep(1)

            # Build the dataframe
            df = pd.concat([df,df_temp])

            # Print progress
            print('Getting reviews ' + str(i+1)+'/ '+str(total_pages))
        print('Complete!!!')

    return df.reset_index().iloc[:,1:]
```

1.4 Examples

1.4.1 Web Scrape one page from tripadvisor

```
In [279]: df,success = webscrape("https://www.tripadvisor.co.uk/Restaurant_Review-g186338-d257
          -Reviews-Cafe_in_the_Crypt-London_England.html",'tripadvisor',False)

Diagnostics: Checking if dataframes are of equal size...
Size: 10
Size: 10
Size: 10
Size: 10
Diagnostics complete!


In [280]: df.head()

Out[280]:                                                 Review  \
          0  Took some Australian relations here during a v...
          1  Visited after walking through Trafalgar Square...
          2  Stopped for lunch before going to the ballet a...
          3  My daughter had the fish and chips. The fish w...
          4  Literally "dropped in" for an afternoon tea to...

                              title  Rating                    date  \
          0                     Novel       5   Reviewed 3 days ago
          1     Quick drink and snack       5   Reviewed 6 days ago
          2         Pre theatre lunch       5   Reviewed 1 week ago
          3  Late lunch disappointing       3   Reviewed 1 week ago
          4    Good for Afternoon Tea       4  Reviewed 2 weeks ago

                                                 fullreview
          0  Took some Australian relations here during a v...
          1  Visited after walking through Trafalgar Square...
          2  Stopped for lunch before going to the ballet a...
          3  My daughter had the fish and chips. The fish w...
          4  Literally "dropped in" for an afternoon tea to...
```

1.4.2 Web Scrape 20 pages from tripadvisor

```
In [281]: inurl1 = "https://www.tripadvisor.co.uk/Restaurant_Review-g186338-d2570383-Reviews"
          inurl2 = "-Cafe_in_the_Crypt-London_England.html"

          df_full = fullscraper(site='tripadvisor',first_url=inurl1+inurl2,url1=inurl1,url2=inu
                                increment_string1="-or",increment_string2="",total_pages=20,\
                                increment=10,output_file='testing.csv')

Getting reviews 0/ 20
Diagnostics: Checking if dataframes are of equal size...
Size: 10
```

```
Size: 10
Size: 10
Size: 10
Diagnostics complete!
Getting reviews 1/ 20
Getting reviews 2/ 20
Getting reviews 3/ 20
Getting reviews 4/ 20
Getting reviews 5/ 20
Getting reviews 6/ 20
Getting reviews 7/ 20
Getting reviews 8/ 20
Error in reading - Re-reading
Getting reviews 9/ 20
Getting reviews 10/ 20
Diagnostics: Checking if dataframes are of equal size...
Size: 10
Size: 10
Size: 10
Size: 10
Diagnostics complete!
Getting reviews 11/ 20
Getting reviews 12/ 20
Getting reviews 13/ 20
Getting reviews 14/ 20
Getting reviews 15/ 20
Getting reviews 16/ 20
Getting reviews 17/ 20
Getting reviews 18/ 20
Getting reviews 19/ 20
Getting reviews 20/ 20
Complete!!!
```

We look at the first 5 rows:

```
In [282]: df_full.head()
```

Out[282]:

	Review	title	Rating	date
0	Took some Australian relations here during a v...	Novel	5	Reviewed 3 days ago
1	Visited after walking through Trafalgar Square...	Quick drink and snack	5	Reviewed 6 days ago
2	Stopped for lunch before going to the ballet a...			
3	My daughter had the fish and chips. The fish w...			
4	Literally "dropped in" for an afternoon tea to...			

```
2         Pre theatre lunch       5   Reviewed 1 week ago
3  Late lunch disappointing       3   Reviewed 1 week ago
4    Good for Afternoon Tea       4  Reviewed 2 weeks ago

                                                fullreview
0  Took some Australian relations here during a v...
1  Visited after walking through Trafalgar Square...
2  Stopped for lunch before going to the ballet a...
3  My daughter had the fish and chips. The fish w...
4  Literally "dropped in" for an afternoon tea to...
```

Looks good. We look at the number of rows and columns:

```
In [283]: df_full.shape

Out[283]: (200, 5)
```

Looks good. We check if there were any read errors (null entries):

```
In [284]: df_full.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
Review        200 non-null object
title         200 non-null object
Rating        200 non-null int64
date          200 non-null object
fullreview    200 non-null object
dtypes: int64(1), object(4)
memory usage: 7.9+ KB
```

Also looks good. We look at the number of unique entries:

```
In [285]: print(df_full['Review'].nunique())
          print(df_full['title'].nunique())

200
199
```

There is a title of a review that is duplicated. We check to see which one that is:

```
In [286]: grp = df_full.groupby('title').count()
          grp[grp['date'] > 1]

Out[286]:         Review  Rating  date  fullreview
          title
          Lovely       2       2     2           2
```

The title that is not unique is 'Lovely'. Let's look at whether this is a legitimate duplicate and not an error:

```
In [287]: df_full[df_full['title'] == 'Lovely']
```

```
Out[287]:                                                  Review   title  Rating  \
          91   A calm quiet place and comparatively inexpensi...  Lovely       5
          149  Beautiful environment. Busy but well stocked a...  Lovely       5

                               date  \
          91    Reviewed 19 June 2018
          149  Reviewed 10 March 2018

                                                  fullreview
          91   A calm quiet place and comparatively inexpensi...
          149  Beautiful environment. Busy but well stocked a...
```

Looks legitimate!

1.5 Web Scrape one page from Yelp

```
In [289]: df,success = webscrape("https://www.yelp.co.uk/biz/ffionas-restaurant-london?\
          osq=Restaurants",'yelp',False)
```

```
Diagnostics: Checking if dataframes are of equal size...
Size: 20
Size: 20
Size: 20
Diagnostics complete!
```

```
In [290]: df.head()
```

```
Out[290]:                                                Review  \
          0  Awesome in all regards. Unpretentious, fun, wa...
          1  We were so happy when we called (last minute) ...
          2  So I honestly picked this place strictly based...
          3  This was our first dinner in London and we did...
          4  Absolute gem! We were so happy when we called ...

                                                   title  Rating        date  \
          0  Awesome in all regards. Unpretentious, fun, wa...       5  30/11/2018
          1  We were so happy when we called (last minute) ...       4   5/1/2019
          2  So I honestly picked this place strictly based...       5  20/11/2018
          3  This was our first dinner in London and we did...       5   8/10/2018
          4  Absolute gem! We were so happy when we called ...       5   5/11/2018

                                                 fullreview
          0  Awesome in all regards. Unpretentious, fun, wa...
```

```
1  We were so happy when we called (last minute) ...
2  So I honestly picked this place strictly based...
3  This was our first dinner in London and we did...
4  Absolute gem! We were so happy when we called ...
```

1.5.1 Web Scrape 10 pages from yelp

```
In [291]: inurl1 = "https://www.yelp.co.uk/biz/ffionas-restaurant-london"
          inurl2 = ""

          df_full = fullscraper(site='yelp',first_url=inurl1+inurl2,url1=inurl1,url2=inurl2,\
                                increment_string1="?start=",increment_string2="",total_pages=1
                                increment=20,output_file='testing.csv')
```

```
Getting reviews 0/ 10
Diagnostics: Checking if dataframes are of equal size...
Size: 20
Size: 20
Size: 20
Diagnostics complete!
Getting reviews 1/ 10
Getting reviews 2/ 10
Getting reviews 3/ 10
Getting reviews 4/ 10
Getting reviews 5/ 10
Getting reviews 6/ 10
Getting reviews 7/ 10
Getting reviews 8/ 10
Getting reviews 9/ 10
Getting reviews 10/ 10
Complete!!!
```

We look at the first 5 rows:

```
In [292]: df_full.head()
```

```
Out[292]:                                                Review  \
          0  Awesome in all regards. Unpretentious, fun, wa...
          1  We were so happy when we called (last minute) ...
          2  So I honestly picked this place strictly based...
          3  This was our first dinner in London and we did...
          4  Absolute gem! We were so happy when we called ...

                                                   title  Rating        date  \
          0  Awesome in all regards. Unpretentious, fun, wa...       5  30/11/2018
          1  We were so happy when we called (last minute) ...       4    5/1/2019
          2  So I honestly picked this place strictly based...       5  20/11/2018
          3  This was our first dinner in London and we did...       5   8/10/2018
```

```
4  Absolute gem! We were so happy when we called ...       5   5/11/2018

                                                  fullreview
0  Awesome in all regards. Unpretentious, fun, wa...
1  We were so happy when we called (last minute) ...
2  So I honestly picked this place strictly based...
3  This was our first dinner in London and we did...
4  Absolute gem! We were so happy when we called ...
```

Looks good. We look at the number of rows and columns:

```
In [293]: df_full.shape

Out[293]: (200, 5)
```

Looks good. We check if there were any read errors (null entries):

```
In [294]: df_full.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
Review        200 non-null object
title         200 non-null object
Rating        200 non-null int64
date          200 non-null object
fullreview    200 non-null object
dtypes: int64(1), object(4)
memory usage: 7.9+ KB
```

Also looks good. We look at the number of unique entries:

```
In [295]: print(df_full['Review'].nunique())
          print(df_full['title'].nunique())

200
200
```

Looks good!!!