A3 Web Scraping

Due 11:59pm Sunday, September 22

For full credit, ensure that you are solving these tasks in the *pandorable* way, in particular you do not want to pull data out of a data frame (or series) and iterate with a loop. As always, conform to our Python style guide.

In this assignment you will explore web scraping. You will (a) write a spider in scrapy (b) scrape information about popular apps from the iTunes store available at https://www.apple.com/uk/itunes/charts/free-apps [1] and (c) perform simple statistical analysis on the data using Pandas. You may find the following sequence of steps to be useful as you work on this part. Of course, these are not the only way to proceed through this part, be free to choose a path that suits your work flow.

1. *Study an example*. This part is just for review. You do not need to submit any file / code from this step.

   Study the solutions in `s1.py`, `s2.py`, and `s3.py` which we saw in class. `s1.py` and `s2.py` do single page scraping. `s3.py` shows how to do multipage scraping: identify the URL pointing to the new page and then transition to that page.

   a. Execute the code with:
      i. `% python -m scrapy runspider s1.py -t csv -o - > s1-out.csv`
      ii. `% python -m scrapy runspider s2.py -t csv -o - > s2-out.csv`
      iii. `% python -m scrapy runspider s3.py -t csv -o - > s3-out.csv`
   b. Check the output in `s1-out.csv`, `s2-out.csv` and `s3-out.csv` for correctness

   In this assignment, you will do the equivalent of `s1.py`, `s2.py` and `s3.py` for the itunes website.

2. *Identify XPath expressions*. Your solution to the assignment will be in the spirit of `s3.py`. Open the page at https://www.apple.com/uk/itunes/charts/free-apps in Chrome and explore the page contents using the Xpath helper plugin and Xpath expressions (refer to the class slides). Identify Xpath expressions for extracting the below from the web page at the above URL. (Note these are names I've given, the actual label names in the HTML are different.)

   a. `app_name`
   b. `category`
   c. `appstore_link_url`
   d. `img_src_url`

   To identify the `XPath` expressions, in addition to Chrome's Xpath helper, you could also consider using the `scrapy` shell:

   ```
   % python -m scrapy shell 'http://www.apple.com/uk/itunes/charts/free-apps/'
   ```

   Once the shell starts, you will have access to the `response` object and can issue XPath queries to it:

   ```
   response.xpath(_____________________)
   ```

3. *Scrape the first page.* Write a spider called `a3-sraja-itunes-topapps-1.py` (of course, replacing `sraja` with your own AndrewID). Ensure that you are able to scrape this information into a CSV file using command line options as shown for `s1.py`. Save the output to `1.csv`.

[1] In the past we have used the US based http://www.apple.com/itunes/charts/free-apps but that site doesn't seem to be currently working and hence we are using the UK based site.

4. *Scrape the the page of a single app*. Pick a single app. For the sake of uniformity, lets all pick the page for WhatsApp. Copy the contents of `a3-sraja-itunes-topapps-1.py` to `a3-sraja-itunes-topapps-2.py`. Edit the `-2.py` file to go directly to the page for WhatsApp and scrape information on star rating as well as the number of ratings by identifying the needed Xpath expressions. Save the output to `2.csv`

5. *Combine the two scrapers.* Again, copy the contents of `a3-sraja-itunes-topapps-1.py` into another file `a3-sraja-itunes-topapps-3.py`. Combine the code in `-1.py` and `-2.py` to scrape multiple pages in the spirit of s3.py. From http://www.apple.com/uk/itunes/charts/free-apps for each application, you will see links to app specific pages (the `appstore_link_url` above). Transition to that page and on each of these app specific pages scrape the star rating as well as the number of ratings.
    e. The star rating
    f. The number of ratings

    Note even after extracting the information you will need to do some cleaning. Do not do the cleaning in your scrapy code. Save what ever you get from the `xpath .extract()` method. Save the output of the scraped information to `3.csv`. This CSV file should now have 4+2 = 6 fields (enumerated from a-f in this write-up).

6. *Analyze the data*. Once you have scraped all the needed information into `3.csv`, you will move onto the analysis phase. Create an Jupyter NB `a3-sraja-itunes-summarize.ipynb` and demonstrate the following analysis using Pandas

    i. Clean the data. Direct scraping will give you pieces of text such as "4.2 out of 5" and "1.2K Ratings". Based on where you scrape, you be able to scrape "4.2" directly instead of "4.2 out of 5". Either is fine.
    ii. Write Pandas code that will add extra columns with cleaned versions of this data e.g., 4.2 and 12000, both stored as numbers. You are welcome to call these new columns as you prefer.
    iii. List the names of the top apps sorted in descending order based on star rating and within those with the same star rating sort based on number of ratings in descending order. If the number of ratings are also the same, sort by app_name in ascending order. Your result should be a data frame with the `app_name`, `star_rating`, and cleaned version of the `num_ratings`.
    iv. For each category list the number of apps. Produce your answer as a series with the app categories as the index.
    v. For each category of app (game, music etc.) list the average rating of all apps in that category and sort in descending order by average rating. Your answer will be a series similar to your answer for iii.
    vi. For each category, list the app with the highest star rating. If there is a tie for apps with the highest star rating, list the one with the greatest number of ratings.

Enter a short description of the various parts (5.i, 5.ii etc.) in markdown cells followed by your analysis using Pandas.

Avoiding Timeouts

After your scrape the main page at https://www.apple.com/uk/itunes/charts/free-apps you will iterate 100 times scraping each individual application. At that time you will hit the website very rapidly and Apple will throttle your access and at times may even shut it off for a few hours. So, it is a good idea to always be nice when scraping. The below is a snippet from my sample solution where you see that I introduce a delay of 1/2 second between each request sent from Scrapy. Apple.com is fine with this. [Note: as always, do not cut/paste code from Word into your programs. Do type it out.]

```
class ItunesSpider(Spider):
    name = "itunes"
    handle_httpstatus_list = [404, 403]
    allowed_domains = ["apple.com"]
    start_urls = ["https://www.apple.com/uk/itunes/charts/free-apps/"]
    custom_settings = { 'DOWNLOAD_DELAY': 0.5 }
```

```
    def parse(self, response):
        apps = response.xpath('...')
        ans=[]
        for app in apps[:5]:
            ...
```

Note that for testing / debugging purposes I only do a few iterations in the for loop. Once you know your code is working correctly, then change the line to

```
for app in apps:
    ...
```

Be sure to use these technics right from the beginning, *before* throttling happens[2]. Given the dynamic nature of the website, it is possible that some links may not work. If you are not able to scrape information about all 100 apps it is okay to miss a few.

Sample csv and output

To guide your efforts, I have provided a sample csv and sample output for my Pandas code. Please note that the itunes website is a live website. So, by the time you scrape it the contents will be slightly different from my csv file. I have provided my output only for testing purposes. When you run your Pandas code with my csv file you should get my sample output. But in the end ensure that your Pandas code runs with the csv file you have created.

What to submit

You will create the following files.

1. a3-sraja-itunes-topapps-1.py
2. 1.csv
3. a3-sraja-itunes-topapps-2.py
4. .csv
5. a3-sraja-itunes-topapps-3.py
6. 3.csv
7. a3-sraja-itunes-summarize.ipynb

Naturally, replace 'sraja' with your AndrewID ☺. Zip all files and submit the zip bundle to Canvas.

[2] Using these techniques after throttling occurs will not help. It would be equivalent to closing the barn door after the horse got away ☺.