Assignment Two

Jacob Berlin

CS432 – Spring 2016

Write a Python program that extracts 1000 unique links from Twitter. You might want to take a look at:

http://thomassileo.com/blog/2013/01/25/using-twitter-rest-api-v1-dot-1-with-python/

But there are many other similar resources available on the web. Note that only Twitter API 1.1 is currently available; version 1 code will no longer work.

Also note that you need to verify that the final target URI (i.e., the one that responds with a 200) is unique. You could have many different shortened URIs for www.cnn.com (t.co, bit.ly, goo.gl, etc.).

You might want to use the search feature to find URIs, or you can pull them from the feed of someone famous (e.g., Tim O'Reilly).

```
from twitter import *
from urlparse import urlparse

unsortedFileName = 'UnsortedURL.txt'

con_secret = "38wZKZuUuwGitHkE3dMpR7jEz"
con_secret_key = "czTV2ryAOTlep7FPC8dVsaAwS28Cw8Z7L8gDCLnj22ioo0uyuG"

token = "2352884547-xheIpcHT0oIjJmzGUkIHwt5X2IZmwogTMh9YWvc"
token_key = "70RS08peFisvJyPOZGlPleQqfg98twYVkIQefeEP1Ifdg"

unsortedFile = open(unsortedFileName, 'a') #Opens the unsorted file

t = Twitter(
    auth=OAuth(token, token_key, con_secret, con_secret_key))

# Get your "home" timeline
x = t.statuses.home_timeline()
#x = t.statuses.user_timeline(screen_name="timoreilly")


twitter_stream = TwitterStream(auth=OAuth(token, token_key, con_secret, con_secret_key))
iterator = twitter_stream.statuses.sample()
counter = 0

for tweet in iterator:
    if 'entities' in tweet:
        for url in tweet['entities']['urls']:
            unsortedFile.write(url['expanded_url'])
            unsortedFile.write("\n")
            #print url['expanded_url']
            print counter
            counter = counter + 1
            if counter == 1000:
                break
    if counter == 1000:
        break

with open('SortedLinks.txt') as f:
    lines = f.readlines()
    lines = [x.strip('\n') for x in lines]

for urls in lines:

    print urls

unsortedFile.close()
```

For this I went through and grabbed a specific amount of URI's from Twitter's main page. For my sort algorithm I got a total of 7,000 to test with so no duplicates would be found. This all came out in an unsorted list, where each URI had the entire line and there were many occurrences of the same URI. (See next page)

http://www.ft.com/intl/cms/s/0/f9e488a2-d115-11e5-831d-09f7778e7377.html#axzz3zuRaGb2z
http://www.wsj.com/articles/google-developing-stand-alone-virtual-reality-headset-1455218948
http://news.samsung.com/us/2016/02/11/samsung-presents-gear-s2-classic-rose-gold-platinum/
http://on.ft.com/20qzHzq
http://buff.ly/20MB6pt
https://twitter.com/tim/status/697581182788108288
https://twitter.com/fttechnews/status/697572960513691648
http://www.parksassociates.com/blog/article/pr-02102016-mwc
https://twitter.com/fttechnews/status/697265308076281858
https://aws.amazon.com/service-terms/
http://om.co/2016/02/09/last-night-at-the-crunchies/
https://vine.co/v/ebQE99eqzMm
http://qz.com/613277
http://www.ft.com/intl/cms/s/0/f9e488a2-d115-11e5-831d-09f7778e7377.html#axzz3zuRaGb2z
http://www.wsj.com/articles/google-developing-stand-alone-virtual-reality-headset-1455218948
http://news.samsung.com/us/2016/02/11/samsung-presents-gear-s2-classic-rose-gold-platinum/
http://on.ft.com/20qzHzq
http://buff.ly/20MB6pt
https://twitter.com/tim/status/697581182788108288
https://twitter.com/fttechnews/status/697572960513691648
http://www.parksassociates.com/blog/article/pr-02102016-mwc
https://twitter.com/fttechnews/status/697265308076281858
https://aws.amazon.com/service-terms/
http://om.co/2016/02/09/last-night-at-the-crunchies/
https://vine.co/v/ebQE99eqzMm
http://qz.com/613277
http://oreil.ly/1o3U8WR
https://twitter.com/arstechnica/status/697797749094379520
https://www.oreilly.com/ideas/agile-isnt-just-for-software-anymore
https://www.coworker.org/petitions/uber-be-fair-increase-fares
http://www.vox.com/2016/2/10/10956978/donald-trump-terrifying?utm_campaign=vox&utm_content=article%3Atop&utm_medium=social&utm_source=twitter
http://www.nytimes.com/2016/02/09/opinion/i-miss-barack-obama.html
http://bit.ly/1PMfp1C
http://bit.ly/1QTtnxI
https://lnkd.in/eAbvSPk
https://twitter.com/planetlabs/status/696834575062839296
https://twitter.com/anthonyonesto/status/696776284895100928
http://m.gulfnews.com/news/uae/government/world-government-summit-live-largest-structural-changes-in-uae-government-approved-1.1667861
https://twitter.com/chriseng/status/696646982669094912
https://twitter.com/redlineracers/status/697951327029227522
http://bit.ly/1Qb1qER
http://vine.co/v/hA1UiIiZe0r
https://vine.co/v/iJhXWhPm1MY
https://twitter.com/kerrywashington/status/697965245713625090
http://fineartamerica.com/featured/bending-light-sarah-loft.html
http://twasul-news.com/b.htm
https://amp.twimg.com/v/742c8808-d626-4cfe-8c75-a2a6f391876d
https://twitter.com/In_A_YamChele/status/697965418179330049
http://TOTO77.GA
http://datv.jp/p000755/
http://voreportovzla.com/2016/02/11/tsj-ordena-comparecer-a-ramos-allup-en-sala-constitucional/

After this step I ran the unsorted list through my condense.py file which took the unsorted list and compared them based off of their net location and then deleted them if they were duplicates:

```
from urlparse import urlparse

unsortedFile = open('UnsortedURL.txt', 'a')
parserFile = open('Parsed.txt', 'w')
condensedFile = open('condensed.txt', 'w')

with open('UnsortedURL.txt') as f:
    lines = f.readlines()
    lines = [x.strip('\n') for x in lines]

for line in lines:
    o = urlparse(line)
    #print o.netloc
    parserFile.write(o.netloc)
    parserFile.write("\n")
parserFile.close()

parserFile = open('Parsed.txt', 'a')
with open('Parsed.txt') as z:
    parser = z.readlines()
    parser = [x.strip('\n') for x in parser]
parserFile.close()
parsing = list(set(parser))
for parsed in parsing:
    print parsed
    condensedFile.write(parsed)
    condensedFile.write("\n")

condensedFile.close()
unsortedFile.close()
```

```
ln.is
youtu.be
www.swarmapp.com
ranking.seeingjp.com
twitter.com
www.ebay.com
nmgam.es
digest.press
tinyurl.com
MB-999.COM
fb.me
www.instagram.com
huff.to
twitter.com
livematchinfo.com
youtu.be
goo.gl
www.instagram.com
goo.gl
du3a.org
ln.is
pbs.twimg.com
www.swarmapp.com
www.instagram.com
twitter.com
bit.ly
www.thebingbing.com
bit.ly
funnybabies.co
bit.ly
straightofficial.com
twitter.com
trendaddicts.cf
twitter.com
twitter.com
gol.am
ameblo.jp
bit.ly
twitter.com
is.gd
iwl.co.jp
scorpioid.bid
maud.autolikers.in
tinyurl.com
ow.ly
WWW.MMMGlobal.BZ
twitter.com
wapo.st
livematchinfo.com
twitter.com
zad-muslim.com
fb.me
```

(Parsed)

```
drive.google.com
me2.do
www.tuttoarte.ch
kca.mundonick.com.br
www.asahi.com
ironna.jp
LONG.LIVE
www.nytimes.com
mery.jp
www.sisatime.co.kr
fleblanc.com
www.sommelierwine.net
www.lantis.jp
blog.livedoor.jp
www.vevo.com
www.muthead.com
cnb.cx
rssfeeds.usatoday.com
splatoon-nawabari.com
nuclearsecrecy.com
konzapata.com
www.englandrugby.com
abcn.ws
streamable.com
www.grandesmedios.com
Goo.gl
www.tuitutil.net
sgoring.blog.me
tomoya39.blog.fc2.com
lineblog.me
sexmare.com
blowjob.fuzzyputtycams.xyz
mixi.jp
www.coworker.org
1000webradios.de
www.antarariau.com
animeanime.jp
yourjewelrycase.com
7asnat.com
www.darkhorse.com
movimail.co
central132.cl
www.facebook.com
HABERTURK.COM
itun.es
www.bygwaah.com
hibarai-arubaito.com
dlove.jp
4NN.cx
allstaresaltoalto.blogspot.com.br
radiobiafra.co
quigs.com
```

At this state, the URI's were finally condensed entirely. With this I had the thousand URI's that were needed to get the time map from. They are all stored in the 'condensed.txt' file and separated by a new line.

Download the TimeMaps for each of the target URIs. We'll use the mementoweb.org
Aggregator, so for example:

URI-R = http://www.cs.odu.edu/

URI-T = http://mementoweb.org/timemap/link/http://www.cs.odu.edu/

You could use the cs.odu.edu aggregator:

URI-T =
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.cs.odu.edu/

But be sure to say which aggregator you use -- they are likely to give different answers.

Create a histogram of URIs vs. number of Mementos (as computed from the TimeMaps). For example, 100 URIs with 0 Mementos, 300 URIs with 1 Memento, 400 URIs with 2 Mementos, etc.

See: http://en.wikipedia.org/wiki/Histogram

Note that the TimeMaps can span multiple pages. Look for links like:

<http://mementoweb.org/timemap/link/1000/http://www.cnn.com/>;rel="timemap";
type="application/link-format"; from ="Sun, 08 Jul 2001 21:30:54 GMT"

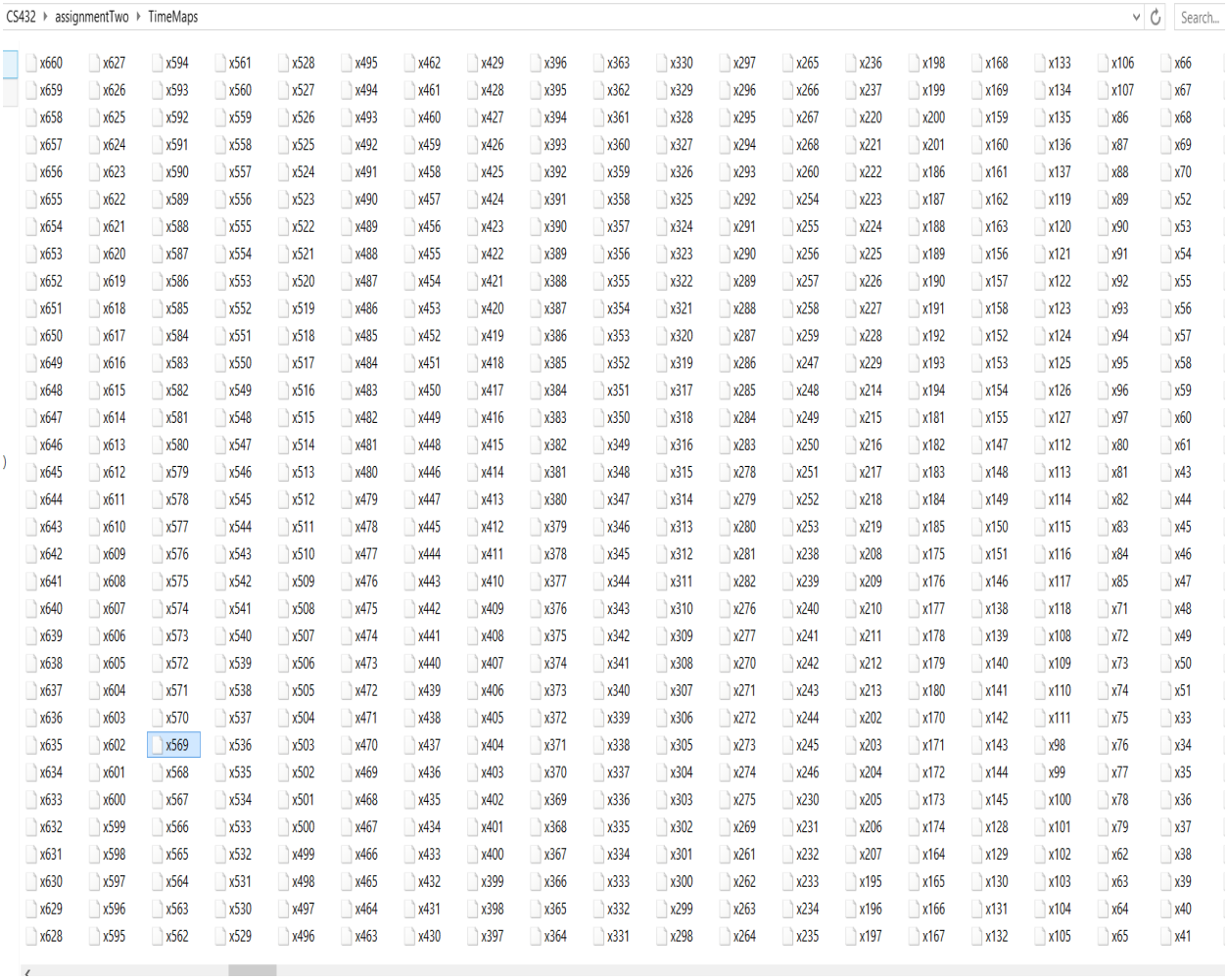
This indicates another page of the TimeMap is available. There can be many pages to a TimeMap.

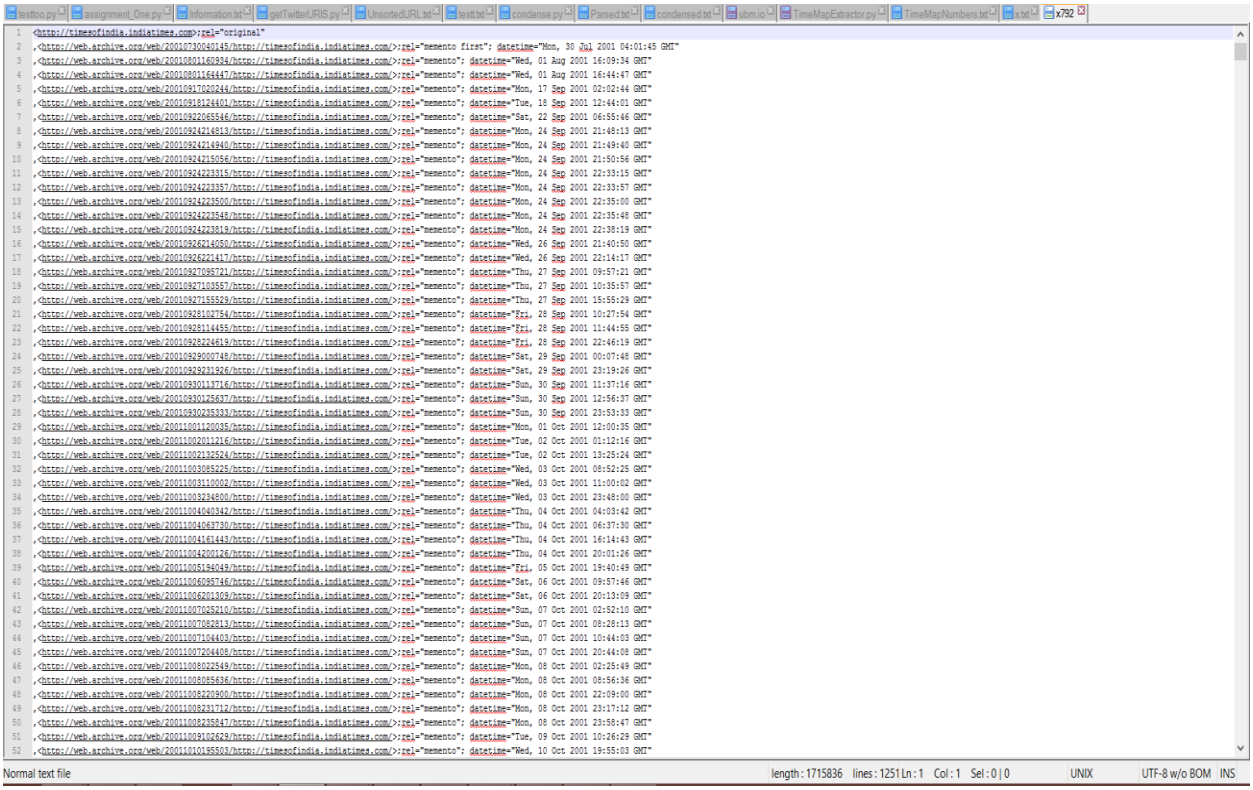
```
#-*- coding: utf-8 -*-

import requests
#import uuid
import os

condensedFile = open('condensed.txt', 'r') #Opens the file w/ all of the uri's to test in read mode
NumbersFile = open('TimeMapNumbers.txt', 'w')
counter = 0
lineNums = 0

URI = "http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/"  #Using the provided momentoproxy

with open('condensed.txt') as f: #goes through the condensed file to grab all of the URI's
    lines = f.readlines()
    lines = [x.strip('\n') for x in lines]

for URLS in lines:
    r = requests.get(URI + 'http://' + URLS)
    print URI + 'http://' + URLS
    print r.status_code
    if r.status_code == 200:
        total = 0
        unique_filename = "TimeMaps/" + "x" + str(counter)
        placeholder = open(unique_filename, 'w')  #opens up the dynamically created file
        placeholder.write(r.text)
        #num_lines = sum(1 for line in open(str(unique_filename)))
        placeholder.close()
        with open(str(unique_filename)) as fileLines: #grabs the amount of momentos from the file
            for lineNums in fileLines:
                found = lineNums.find('archive')
                if found != -1 and found != 0:
                    total += 1
        print total
        NumbersFile.write(str(total)) #adds the total momentos for the single timemap to the NumbersFile
        NumbersFile.write("\n")
        counter = counter + 1


condensedFile.close()
NumbersFile.close()
```

For this part of the project I created a new python program called TimeMapExtractor which grabbed all of the URI's that I had formatted in the previous problem and ran each of them through the supplied cs.odu.edu momentoproxy URI. The program would make sure that each of the URI's gave a '200' response code before creating a new file and putting the time map data inside of it. During the compilation step I made sure to add another loop to extract the amount of momentos that each time map had while it was open and stored those numbers in the TimeMapNumbers.txt. When the URI would give a response that was not a 200, I had the program add in a '0' for the amount of momentos that the time map had.

```
$ python TimeMapExtractor.py
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://Fortune420.com
200
15
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.comicw.co.kr
404
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://go.shr.lc
200
51
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://icogr.com
200
1
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://fifacoinshq.com
404
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://yoldaolmak.com
200
181
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://quad.aldub.online
404
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.alofokemusic.net
200
468
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.popularwoodworking.com
200
928
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://nuascannan.com
404
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://vb.bonofa.com
200
2
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://bijosokuhou2ch.blog.fc2.com
200
1
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.crowdfireapp.com
200
208
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.bukalapak.com
200
630
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://quiboo.cl
200
5
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://padsexy.sakuratan.com
200
3
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.designyourway.net
200
444
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://glbn.ca
200
17
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://de.specsen.com
200
1
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://ranking.seeingjp.com
404
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://wapo.st
200
850
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.city.osaka.lg.jp
200
311
http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://press.bandai.com
```

The compilation of TimeMapExtractor.py will show the momentoproxy link along with the response code and the amount of momentos in the file.

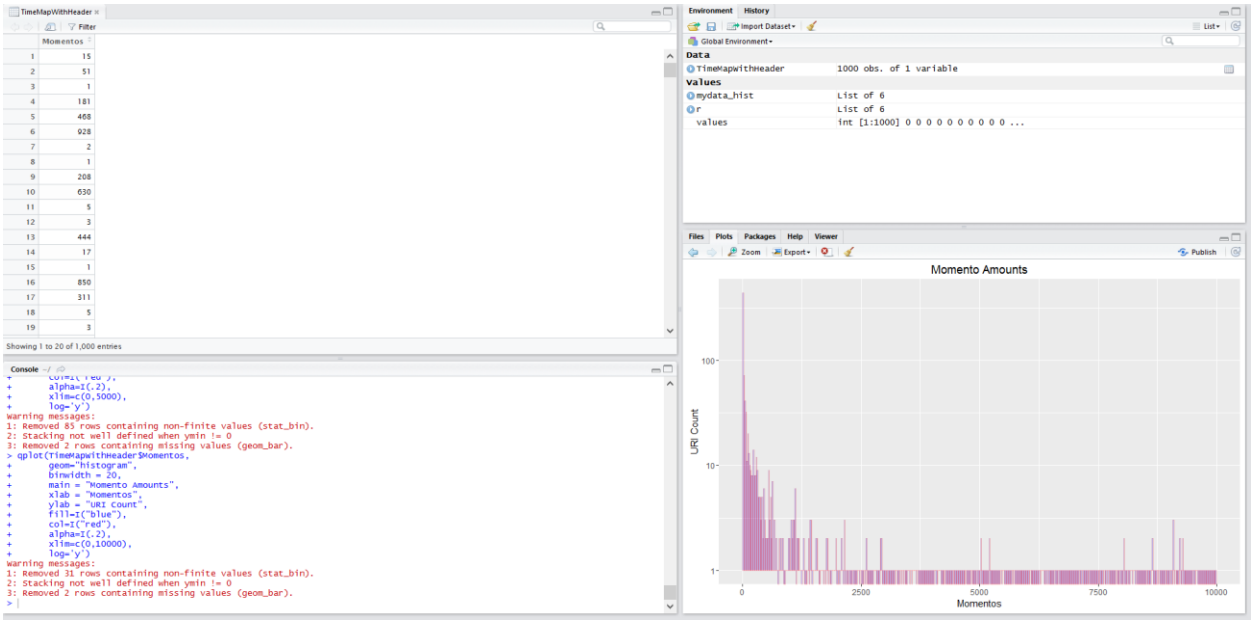

CS432 ▸ assignmentTwo ▸ TimeMaps
Search...
x660 x627 x594 x561 x528 x495 x462 x429 x396 x363 x330 x297 x265 x236 x198 x168 x133 x106 x66
x659 x626 x593 x560 x527 x494 x461 x428 x395 x362 x329 x296 x266 x237 x199 x169 x134 x107 x67
x658 x625 x592 x559 x526 x493 x460 x427 x394 x361 x328 x295 x267 x220 x200 x159 x135 x86 x68
x657 x624 x591 x558 x525 x492 x459 x426 x393 x360 x327 x294 x268 x221 x201 x160 x136 x87 x69
x656 x623 x590 x557 x524 x491 x458 x425 x392 x359 x326 x293 x260 x222 x186 x161 x137 x88 x70
x655 x622 x589 x556 x523 x490 x457 x424 x391 x358 x325 x292 x254 x223 x187 x162 x119 x89 x52
x654 x621 x588 x555 x522 x489 x456 x423 x390 x357 x324 x291 x255 x224 x188 x163 x120 x90 x53
x653 x620 x587 x554 x521 x488 x455 x422 x389 x356 x323 x290 x256 x225 x189 x156 x121 x91 x54
x652 x619 x586 x553 x520 x487 x454 x421 x388 x355 x322 x289 x257 x226 x190 x157 x122 x92 x55
x651 x618 x585 x552 x519 x486 x453 x420 x387 x354 x321 x288 x258 x227 x191 x158 x123 x93 x56
x650 x617 x584 x551 x518 x485 x452 x419 x386 x353 x320 x287 x259 x228 x192 x152 x124 x94 x57
x649 x616 x583 x550 x517 x484 x451 x418 x385 x352 x319 x286 x247 x229 x193 x153 x125 x95 x58
x648 x615 x582 x549 x516 x483 x450 x417 x384 x351 x317 x285 x248 x214 x194 x154 x126 x96 x59
x647 x614 x581 x548 x515 x482 x449 x416 x383 x350 x318 x284 x249 x215 x181 x155 x127 x97 x60
x646 x613 x580 x547 x514 x481 x448 x415 x382 x349 x316 x283 x250 x216 x182 x147 x112 x80 x61
x645 x612 x579 x546 x513 x480 x446 x414 x381 x348 x315 x278 x251 x217 x183 x148 x113 x81 x43
x644 x611 x578 x545 x512 x479 x447 x413 x380 x347 x314 x279 x252 x218 x184 x149 x114 x82 x44
x643 x610 x577 x544 x511 x478 x445 x412 x379 x346 x313 x280 x253 x219 x185 x150 x115 x83 x45
x642 x609 x576 x543 x510 x477 x444 x411 x378 x345 x312 x281 x238 x208 x175 x151 x116 x84 x46
x641 x608 x575 x542 x509 x476 x443 x410 x377 x344 x311 x282 x239 x209 x176 x146 x117 x85 x47
x640 x607 x574 x541 x508 x475 x442 x409 x376 x343 x310 x276 x240 x210 x177 x138 x118 x71 x48
x639 x606 x573 x540 x507 x474 x441 x408 x375 x342 x309 x277 x241 x211 x178 x139 x108 x72 x49
x638 x605 x572 x539 x506 x473 x440 x407 x374 x341 x308 x270 x242 x212 x179 x140 x109 x73 x50
x637 x604 x571 x538 x505 x472 x439 x406 x373 x340 x307 x271 x243 x213 x180 x141 x110 x74 x51
x636 x603 x570 x537 x504 x471 x438 x405 x372 x339 x306 x272 x244 x202 x170 x142 x111 x75 x33
x635 x602 x569 x536 x503 x470 x437 x404 x371 x338 x305 x273 x245 x203 x171 x143 x98 x76 x34
x634 x601 x568 x535 x502 x469 x436 x403 x370 x337 x304 x274 x246 x204 x172 x144 x99 x77 x35
x633 x600 x567 x534 x501 x468 x435 x402 x369 x336 x303 x275 x230 x205 x173 x145 x100 x78 x36
x632 x599 x566 x533 x500 x467 x434 x401 x368 x335 x302 x269 x231 x206 x174 x128 x101 x79 x37
x631 x598 x565 x532 x499 x466 x433 x400 x367 x334 x301 x261 x232 x207 x164 x129 x102 x62 x38
x630 x597 x564 x531 x498 x465 x432 x399 x366 x333 x300 x262 x233 x195 x165 x130 x103 x63 x39
x629 x596 x563 x530 x497 x464 x431 x398 x365 x332 x299 x263 x234 x196 x166 x131 x104 x64 x40
x628 x595 x562 x529 x496 x463 x430 x397 x364 x331 x298 x264 x235 x197 x167 x132 x105 x65 x41



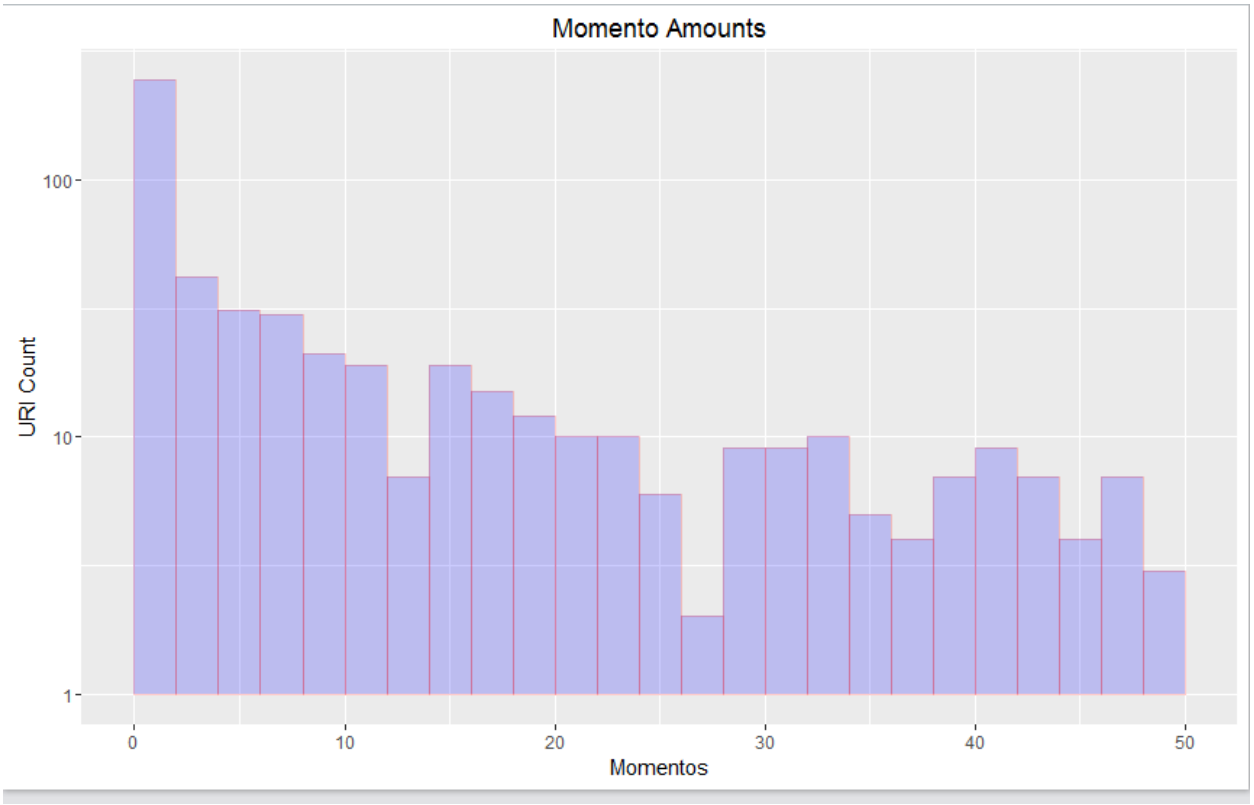
Now, out of the total 1,000 URI's I started with, a total of 792 ran through the extractor and had a specific amount of momentos. For the other 208 URI's, a '0' was added into the numbers file (shown below).

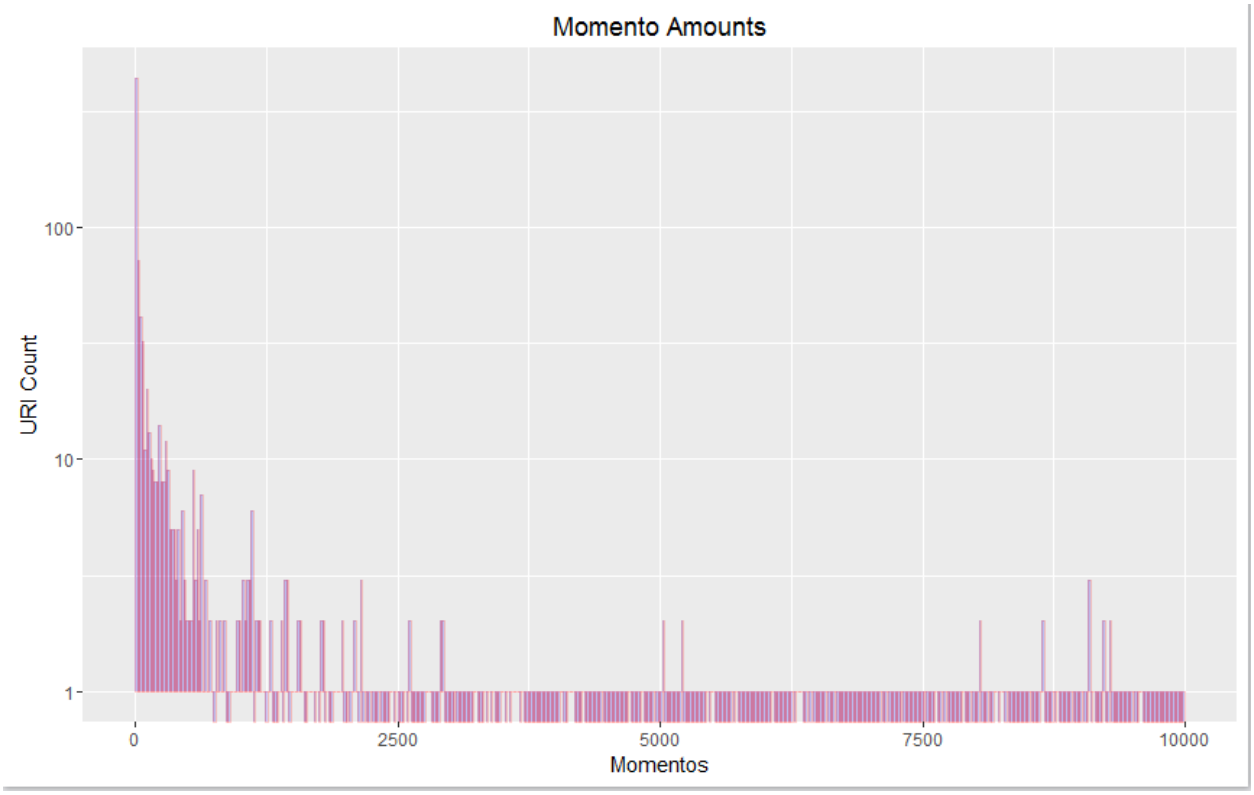
```
385  1
386  2441
387  69
388  8
389  9561
390  634
391  1972
392  17
393  3
394  602
395  991
396  308
397  4452
398  1
399  24
400  230
401  445
402  402
403  22
404  21
405  2795
406  86
407  2619
408  255
409  14
410  1035
411  632
412  3
413  6
414  80
415  43286
416  17195
417  72
418  8
419  146
420  519
421  1915
422  7
423  1
424  9
425  3693
426  54144
427  1
428  1103
429  145
430  21569
431  102
432  265
433  2430
434  636
435  4
436  21892
```

Now that I have all of the time map data, I feed the data into R Studio to create the histogram.



After using R Studio to put the data inside of a histogram, I was able to limit the data to two different histograms to model the amount of URI possibilities that I received. This was using the ggplot library built into R Studio.




Momento Amounts
URI Count
100
10
1
0
2500
5000
7500
10000
Momentos

3. Estimate the age of each of the 1000 URIs using the "Carbon Date" tool:

http://ws-dl.blogspot.com/2014/11/2014-11-14-carbon-dating-web-version-20.html

Note: you'll should download the library and run it locally; don't try to use the web service.

For URIs that have > 0 Mementos and an estimated creation date, create a graph with age (in days) on one axis and number of mementos on the other.

Not all URIs will have Mementos, and not all URIs will have an estimated creation date. State how many fall into either categories.

```
#-*- coding: utf-8 -*-

import requests
#import uuid
import os
import datetime
from dateutil import parser


n = datetime.datetime.now()
empty_string = ""
date_format = "%Y-%m-%dT%H:%M:%S"
new_date_format = "%Y-%m-%d"
condensedFile = open('condensed.txt', 'r') #Opens the file w/ all of the uri's to test in read mode
counter = 0
age = 0

ScriptName = "python local.py"  #Using the provided momentoproxy

with open('condensed.txt') as f: #goes through the condensed file to grab all of the URI's
    lines = f.readlines()
    lines = [x.strip('\n') for x in lines]

for URLS in lines:
    r = os.system(ScriptName + ' http://' + URLS)
    print ScriptName + ' http://' + URLS
    #num_lines = sum(1 for line in open(str(unique_filename)))
    #NumbersFile.write(str(total)) #adds the total momentos for the single timemap to the NumbersFile
    #NumbersFile.write("\n")
    #counter = counter + 1

AgeFile = open('URLAgeNumbers.txt', 'a')
FinalAgeFile = open('FinalAgeFile.txt', 'a')
with open('URLAgeNumbers.txt') as z:
    Age = z.readlines()
    Age = [x.strip('\n') for x in Age]

for cAges in Age:
    if cAges != empty_string:
        y = datetime.datetime.strptime(cAges, date_format)
        p = n.date() - y.date()
        a = str(p).split("days", 1)[0]
        FinalAgeFile.write(a)
        FinalAgeFile.write("\n")
    #print cAges


condensedFile.close()
AgeFile.close()
FinalAgeFile.close()
```

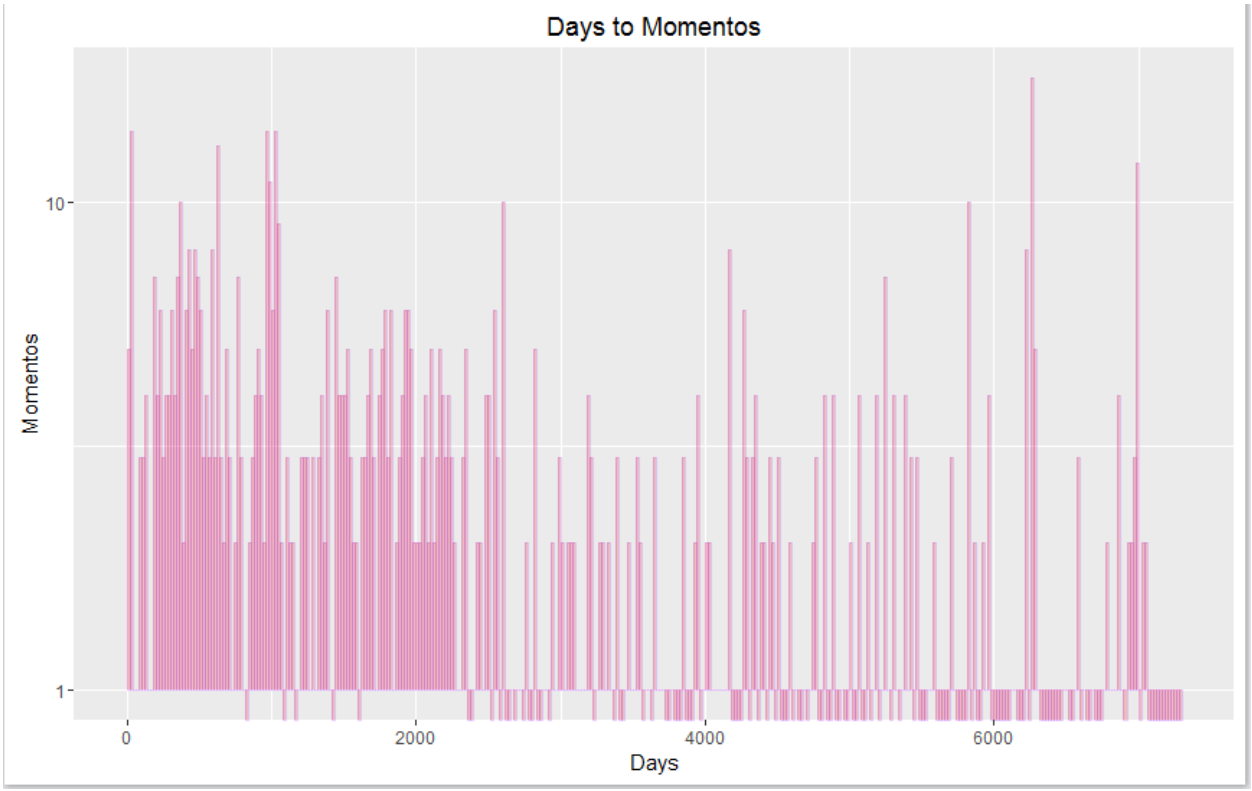
To get the list of CarbonDates for each of the URI's that I retrieved in the previous questions I downloaded the CarbonDate tool and ran it locally. I passed in each of the URI's individually, got the earliest date to which it was created, and parsed the data to just show the amount of days since the creation.

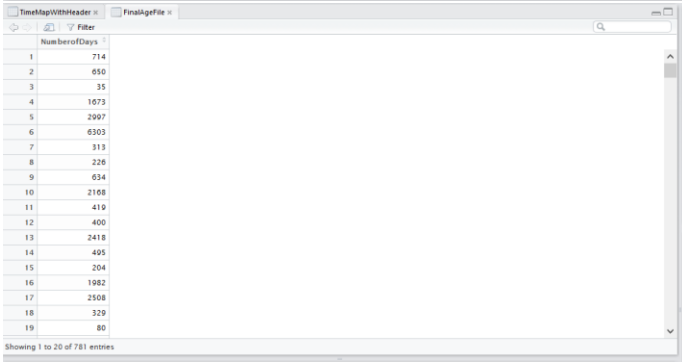
```
python local.py http://www.ttf13.com
Done Last Modified
<type 'exceptions.ValueError'> getBitly.py 41
Done Bitly
Done Google
Done Archives 1
runtime in seconds:  30
{
  "URI": "http://www.kbookmark.com\r",
  "Estimated Creation Date": "2004-04-06T01:18:04",
  "Last Modified": "",
  "Bitly.com": "",
  "Topsy.com": "Topsy is out of service",
  "Backlinks": "",
  "Google.com": "",
  "Archives": [
    [
      "Earliest",
      "2004-04-06T01:18:04"
    ],
    [
      "By_Archive",
      {
        "http://www.kbookmark.com/": "2004-04-06T01:18:04"
      }
    ]
  ]
}

python local.py http://www.kbookmark.com
Done Last Modified
<type 'exceptions.ValueError'> getBitly.py 41
Done Bitly
Done Google
Done Archives 1
runtime in seconds:  30
{
  "URI": "http://www.homedeliverydriverjobs.com\r",
  "Estimated Creation Date": "2012-12-21T10:30:24",
  "Last Modified": "",
  "Bitly.com": "",
  "Topsy.com": "Topsy is out of service",
  "Backlinks": "",
  "Google.com": "",
  "Archives": [
    [
      "Earliest",
      "2012-12-21T10:30:24"
    ],
    [
      "By_Archive",
      {
        "http://www.homedeliverydriverjobs.com/": "2012-12-21T10:30:24"
      }
    ]
  ]
}

python local.py http://www.homedeliverydriverjobs.com
Done Last Modified
<type 'exceptions.ValueError'> getBitly.py 41
Done Bitly
Done Google
Done Archives 1
```

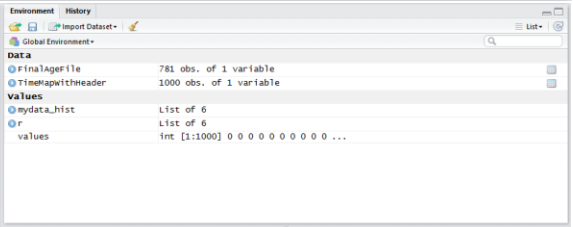
```
562  2006-06-03T17:45:49
563  2001-09-22T09:10:18
564  2000-01-11T02:39:51
565  2001-09-24T15:04:22
566  2005-04-25T05:52:40
567  2007-03-29T07:51:48
568  2007-06-29T13:55:39
569  2003-10-11T08:11:25
570  2013-07-21T18:05:21
571  2001-10-13T10:58:45
572  2009-08-31T09:33:30
573  2016-01-09T01:04:21
574  2003-12-08T11:34:03
575
576  2012-10-11T04:45:10
577  1997-06-13T22:20:18
578  2001-10-01T14:05:09
579
580  2011-09-03T12:11:11
581  1997-07-26T03:06:55
582
583
584  2011-04-16T16:42:21
585
586  2013-04-02T15:18:50
587
588  2013-08-13T03:04:07
589  2013-12-30T02:18:00
590  1996-11-13T19:18:58
591  2011-11-20T16:43:16
592  2011-12-13T13:38:40
593  2005-08-01T19:39:45
594  1999-11-18T00:06:55
595  2012-02-07T02:31:44
596
597  1998-12-02T07:48:16
598  2003-02-04T21:03:26
599  2011-03-28T19:06:22
600  1997-10-11T21:47:49
601
602  2015-07-02T22:48:56
603  2012-03-28T22:59:32
604  2009-02-10T05:18:57
605
606
607
608  2011-02-02T06:30:03
609
610
611  2013-12-09T21:04:00
612  2011-05-06T16:55:46
613  2008-09-15T16:42:23
```

```
678  2257
679  342
680  375
681  1905
682  1550
683  1996
684  6273
685  4504
686  66
687  4264
688  6979
689  961
690  5828
691  4760
692  570
693  173
694  6273
695  6993
696  196
697  2493
698  1530
699  5191
700  5073
701  1329
702  541
703  3312
704  5461
705  5921
706  134
707  1959
708  1456
709  33
710  317
711  702
712  4014
713  586
714  2488
715  1539
716  6273
717  4461
718  5828
719  6
720  4404
721  3139
722  6995
723  244
724  2073
725  5192
726  6988
727  211
728  6862
729  1969
```

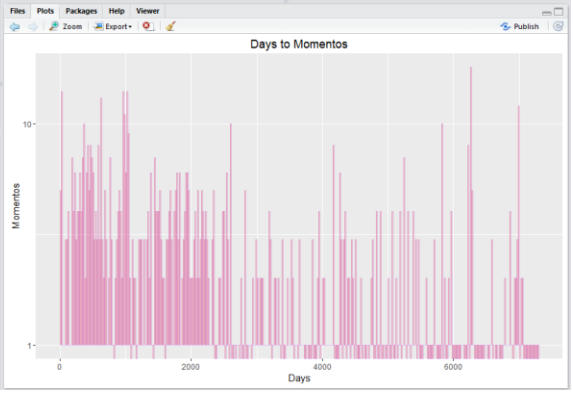
Finally, with the dates received I compared them to the momentos grabbed in question 2 and created a graph in R comparing the two.


Days to Momentos
Momentos
10
1
0
2000
4000
6000
Days


TimeMapWithHeader
FinalAgeFile
Filter
NumberofDays
1 714
2 650
3 35
4 1673
5 2997
6 6303
7 313
8 226
9 634
10 2168
11 419
12 400
13 2418
14 495
15 204
16 1982
17 2508
18 329
19 80
Showing 1 to 20 of 781 entries


Environment
History
Import Dataset
Global Environment
Data
FinalAgeFile 781 obs. of 1 variable
TimeMapWithHeader 1000 obs. of 1 variable
values
mydata_hist List of 6
r List of 6
values int [1:1000] 0 0 0 0 0 0 0 0 0 0 ...


Console
fill=I("green"),
col=I("yellow"),
alpha=I(.2),
xlim=c(0,10000),
log='y')
Warning messages:
1: Stacking not well defined when ymin != 0
2: Removed 2 rows containing missing values (geom_bar).
> qplot(FinalAgeFile$NumberofDays,
geom="histogram",
binwidth = 20,
main = "Days to Momentos",
xlab = "Days",
ylab = "Momentos",
fill=I("red"),
col=I("purple"),
alpha=I(.2),
xlim=c(0,7300),
log='y')
Warning messages:
1: Stacking not well defined when ymin != 0
2: Removed 2 rows containing missing values (geom_bar).


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Days to Momentos
Momentos
10
1
0
2000
4000
6000
Days