Introduction to Digital Libraries: Assignment #1

Dr. Nelson

Alexander Nwala

Thursday, February 12, 2015

Contents

Problem 1

Write a program that extracts 10000 tweets with links from Twitter. Reference: http://thomassileo.com/blog/2013/01/25/using-twitter-rest-api-v1-dot-1-with-python/. Other similar resources are available.
Note that only Twitter API 1.1 is currently available; version 1 code will no longer work.

Save the tweet URIs, and the mapping to the link(s) each tweet contains Note: tweets can have >1 links
For each t.co link, use

```
"curl -I -L"
```

to record the HTTP headers all the way to a terminal HTTP status (i.e. chase down all the redirects)

How many unique final URIs? How many duplicate URIs? Build a histogram of how many redirects (every URI will have at least 1) http://en.wikipedia.org/wiki/Histogram Build a histogram of HTTP status codes encountered (you'll have at least 20000: 10000 301s, and 10000+ more)

SOLUTION

The solution for this problem is outlined by the following steps:

1. **Extract Tweets:** This was achieved by utilizing Tweepy's [2] Twitter search API. As outlined by Listing 1. the tweets were extracted using the query:

   ```
   http://www.
   ```

   To extract 30 tweets which fulfill the search criteria. However, in order not to exceed Twitter's rate limiting on search [1], Listing 1. sleeps in between requests.

Listing 1: Extract Tweets

```
#for tweet in tweepy.Cursor(api.search, q=searchQuery).items():
#for tweet in tweepy.Cursor(api.search, q=searchQuery, since="2014-01-01",until
    ="2014-09-19").items(15):
#for tweet in tweepy.Cursor(api.search, q=searchQuery, since_id=long(sinceIDValue)
    ).items(15):
for tweet in tweepy.Cursor(api.search, q=searchQuery).items(30):

     localTimeTweet = datetime_from_utc_to_local(tweet.created_at)
     #print tweet.id, tweet.created_at
     if( tweet.id > sinceIDValue ):
          sinceIDValue = tweet.id

     expandedUrlsList = getUrisArray(tweet.entities['urls'])

     if(len(expandedUrlsList) > 0):

          for u in expandedUrlsList:
               if(len(u) > 0):
                    u = u.lower().strip()
```

Problem 1 continued on next page. . .

```
                    print "...adding: ", tweet.id, u, localTimeTweet
                    urlsDataFile.write(str(tweet.id) + ', ' + u + ', ' + str(
                        localTimeTweet) + '\n' )
```

2. **Follow redirects, count redirects, and record HTTP status codes:** Given that Twitter shortens URIs embedded in Tweets, Listing 2. resolves redirections by continuously retrieving the URIs in the Location attribute of the 301 responses. This is done until the final URI is retrieved. Listing 2. also counts the number of redirections and stores the HTTP status codes simultaneously.

Listing 2: Extract Tweets

```
def followAndCountTheRedirect(url):
     url = url.strip()
     redirectionCount = 0
     if( len(url) > 0 ):

          indexOfLocation = 0
          httpResponseCodes = ''
          while indexOfLocation > -1:

               co = 'curl -s -I ' + url
               output = commands.getoutput(co)

               indexOfFirstNewLine = output.find('\n')
               if( indexOfFirstNewLine > -1 ):
                    httpResponseCodes = output[0:indexOfFirstNewLine].split(' ')
                        [1] + ' ' + httpResponseCodes

               #if( len(httpResponseCodes) > 0 ):
               #    httpResponseCodes = httpResponseCodes[:-1]

               indexOfLocation = output.find('location:')
               if( indexOfLocation == -1 ):
                    indexOfLocation = output.find('Location:')


               if( indexOfLocation > -1 ):
                    indexOfNewLine = output.find('\n', indexOfLocation + 9)
                    url = output[indexOfLocation + 9:indexOfNewLine]
                    url = url.strip()
                    redirectionCount = redirectionCount + 1

     return redirectionCount, url, httpResponseCodes
```

```
The file originalLinksFile.txt contains the complete data of form:

<
TWEET ID: Unique Identifier for a Tweets
URI: Final URI derived by following the redirects
REDIRECTION COUNT: The number of redirections
[REDIRECTION CODES]: A list of HTTP response status codes seen while
```

```
following redirects
TWEET CREATED AT DATETIME: The timestamp on Tweets
>
```

3. **Count unique URIs:** Based on Listing 3. This statistic was collected: out of 10,000 Tweets collected, 9,442 tweets were unique, however, only 2,165 URIs were unique. This was due to overlapping search results.

Listing 3: Count Unique URIs

```
def countDuplicatesAndMakeUniqueLinksFile(inputFileName):
     inputFileName = inputFileName.strip()
     if( len(inputFileName) > 0 ):

          lines = []
          try:
               inputFile = open(inputFileName.strip(), 'r')
               lines = inputFile.readlines()
               inputFile.close()
          except:
               exc_type, exc_obj, exc_tb = sys.exc_info()
               fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
               print(fname, exc_tb.tb_lineno, sys.exc_info() )

          uniqueListOfURIs = []
          dataToWriteToFile = []
          #URI: <TWEET ID, URI, REDIRECTION COUNT, [REDIRECTION CODES], TWEET
              CREATED AT>
          for URI in lines:
               tuples = URI.strip().split(', ')

               if( tuples[1] not in uniqueListOfURIs ):
                    uniqueListOfURIs.append(tuples[1])
                    dataToWriteToFile.append(URI.decode('ascii', 'ignore'))

          try:
               outputFile = open('unique_'+inputFileName, 'w')
               outputFile.writelines(dataToWriteToFile)
               outputFile.close()
          except:
               exc_type, exc_obj, exc_tb = sys.exc_info()
               fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
               print(fname, exc_tb.tb_lineno, sys.exc_info() )
```

4. **Draw histograms:** Due to Listing 4. Chart 1. summarizes the distribution of HTTP status codes. and Chart 2. summarizes the distribution of redirection counts.

Chart 1: Distribution of HTTP status codes

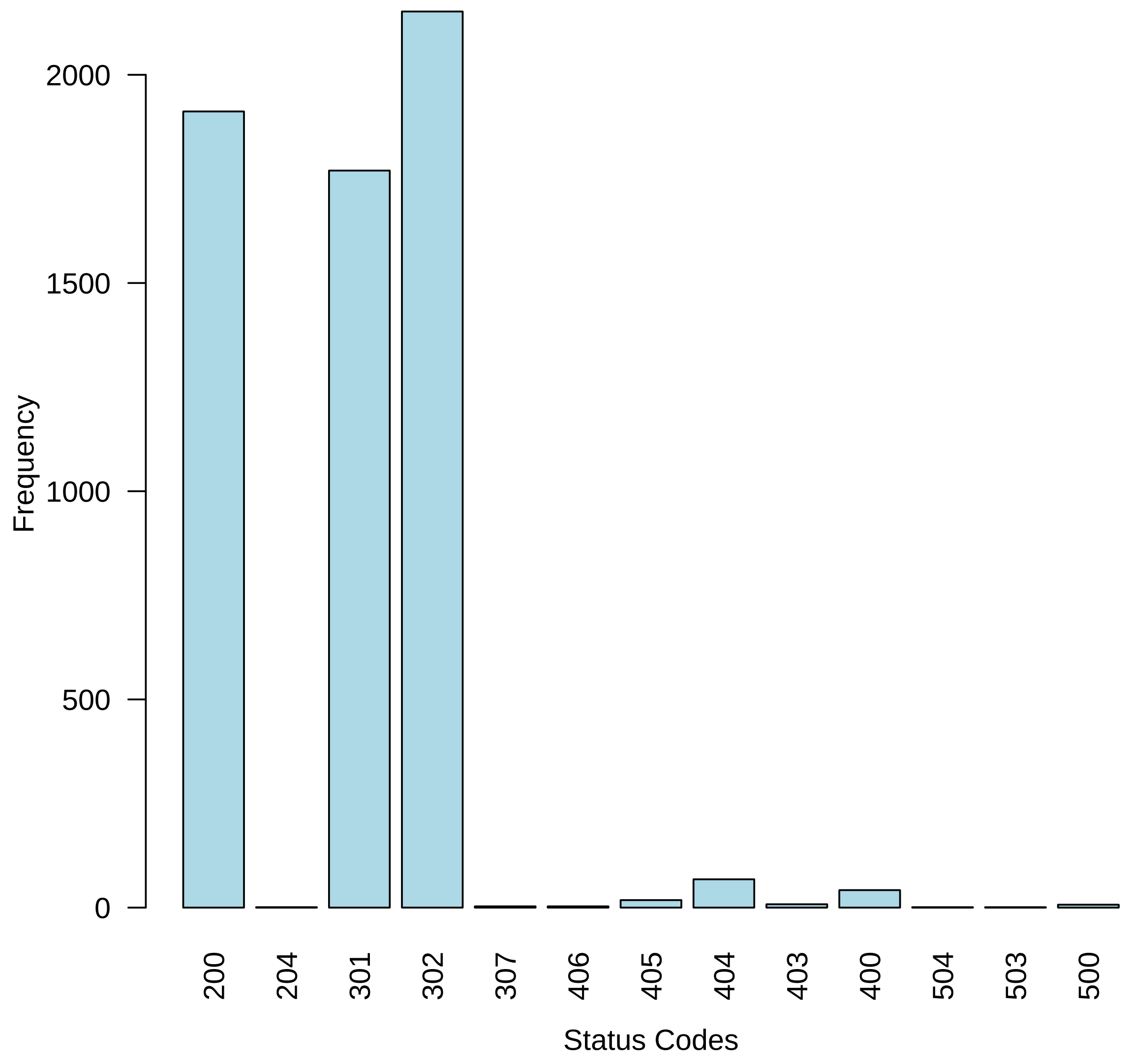
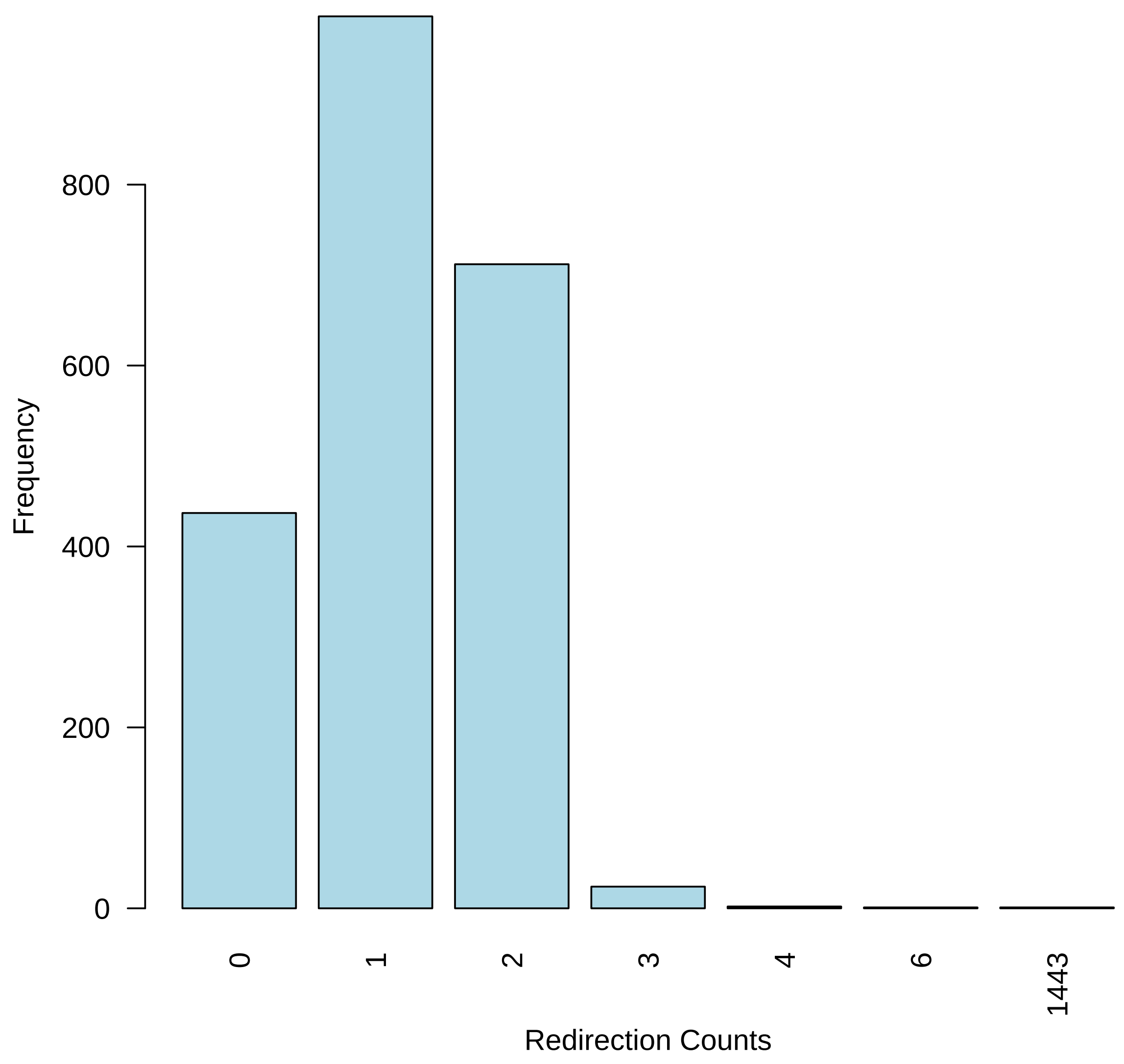


Chart 2: Distribution of Redirection Counts



Listing 4: Draw Histograms

```
#!/usr/bin/env Rscript

rawInputStatusCodes <- read.table('STATUS_CODE_FREQUENCY.txt', header=T)
rawInputRedirectionCounts <- read.table('REDIRECTION_COUNT_FREQUENCY.txt', header=
    T)
rawInputAge <- read.table('DELTA_DAYS.txt', header=T)

binsStatusCodes <-
c(
200,
204,
301,
302,
307,
406,
405,
404,
403,
400,
504,
503,
500
)

binsRedirectionCounts <-
c(
0,
1,
2,
3,
4,
6,
1443
)

barplot( rawInputStatusCodes$STATUS_CODE_FREQUENCY, las=2, names=binsStatusCodes,
    xlab="Status Codes", ylab="Frequency", main="Chart 1: Distribution of HTTP
    status codes", col="lightblue")
barplot( rawInputRedirectionCounts$REDIRECTION_COUNT_FREQUENCY, las=2, names=
    binsRedirectionCounts, xlab="Redirection Counts", ylab="Frequency", main="
    Chart 2: Distribution of Redirection Counts", col="lightblue")
hist( rawInputAge$AGE_DIFFS_DAYS, xlab="Age Values (days)", ylab="Frequency", main
    ="Chart 3: Distribution of Age (TweetDateTime - Est. DateTime)", col="
    lightblue")

mean( rawInputAge$AGE_DIFFS_DAYS )
median( rawInputAge$AGE_DIFFS_DAYS )
sd( rawInputAge$AGE_DIFFS_DAYS )
std <- function(x) sd(x)/sqrt(length(x))
std(rawInputAge$AGE_DIFFS_DAYS)
```

The file unique_originalLinksFile.txt contains data in the same format

Problem 1 continued on next page. . .

```
        as originalLinksFile.txt, but does not contain duplicates
```

Problem 2

Use "Carbon Date" to estimate the age of each link(s) in a tweet See: http://ws-dl.blogspot.com/2013/04/2013-04-19-carbon-dating-web.html
Create a histogram of ($\text{Age}_{\text{tweet}}$ - Age_{link}) Many (most?) deltas will be 0, but there should be many >0 For these deltas, compute: median, mean, std dev, std err. Use wget to download the text for all the links. Hold on to those, well come back to them later. See:

http://superuser.com/questions/55040/save-a-single-web-page-with-background-images-with-wget
http://stackoverflow.com/questions/6348289/download-a-working-local-copy-of-a-webpage

SOLUTION

The solution for this problem is outlined by the following steps:

1. **Carbon date URIs:** This was achieved by https://github.com/HanySalahEldeen/CarbonDate The estimated creation date of the URIs from **unique_originalLinksFile.txt** was written into **final_cd_unique_unique_originalLinksFile.txt**. Below is a summary are the summary statistics derived (Listing 4.) from the the difference between the Tweets age and the estimated creation date from Carbon Date for 1,684 URIs.

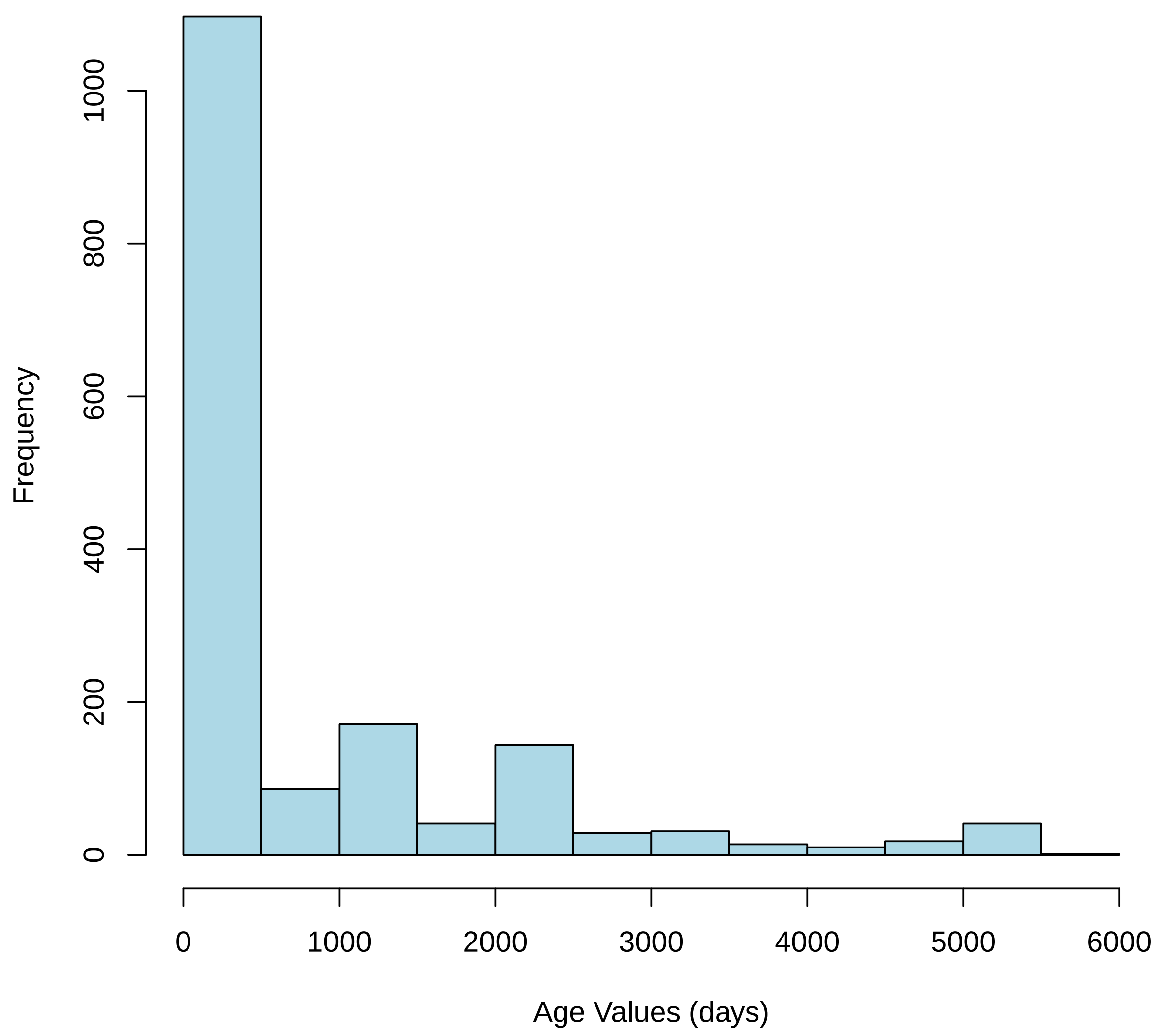
   ```
   Mean: 774.7267 days
   Median: 58 days
   Standard Deviation: 1250.453
   Standard Error: 30.48073
   ```

   Based on the foregoing data, it can be seen that the age values is highly skewed to the left (Approaching 0), probably because the links in many Tweets are new, hence unlikely to have been archived.

2. **Draw histogram:** Chart 3. due to Listing 4. outlines a histogram of $\text{Age}_{\text{tweet}}$ - Age_{link} entries from **DELTA_DAYS.txt** derived from **final_cd_unique_unique_originalLinksFile.txt**

Problem 2 continued on next page. . .

Chart 3: Distribution of Age (TweetDateTime − Est. DateTime)



3. **Download HTML text of links:** Based on input from **unique_originalLinksFile.txt**, Listing 5. utilizes curl to download and save the HTML text of the links into the folder RawHtml, the file names were derived from a hash calculated from the respective URIs.

Listing 5: Download HTML

```
def extractHTMLAndSave():

     if( len(uriHashDictionary) > 0 ):
          count = 1
          for uri in uriHashDictionary:

               filename =  str(count) + '-' + uriHashDictionary[uri] + '.html'
               co = 'curl -s -L ' + uri + ' > ./RawHtml/' + filename

               commands.getoutput(co)
               print '...saved', filename, count
               count = count + 1
```

References

[1] API Rate Limits. https://dev.twitter.com/rest/public/rate-limiting. Accessed: 2015-02-08.

[2] Tweepy. http://tweepy.readthedocs.org/en/v3.2.0/. Accessed: 2015-02-08.