Healthcare Twitter Analysis

Working Document

George Fisher [*]

original publication October 23, 2014

Abstract

'Healthcare Twitter Analysis' is an Open Source project to investigate ways to improve the quality of medical care with Data Science techniques applied to Twitter data. The project began with two handicaps: (1) the data was inadequate and (2) there was little or no understanding on the part of the crowd-sourced team of the analytical tools available. This paper details the efforts of one participant to solve these problems and to provide a base upon which others could build.

The project website is [Mehta and Saama Technologies, 2013]; the GitHub repository for this paper can be found at [Fisher, 2014b].

NOTE: April 8, 2016 changes have been made to reflect a change in AWS S3 data storage

Contents

[*]George escaped from a 30-year international life of crime on Wall Street, got a Masters degree in quantitative finance from MIT, climbed Kilimanjaro and (some of) Everest and currently is pursuing an interest in machine learning which was originally motivated by Bill Howe's fantastic Coursera course *Introduction to Data Science*. george at georgefisher dot com. Please see https://github.com/grfiv/healthcare_twitter_analysis

1 Summary

The Healthcare Twitter Analysis Project has as its objective finding ways to use Twitter data to help in the provision of health care, whether in the area of medical research, in helping in the daily work of professionals engaged in the field, in advancing public policy, etc.

The project began with over 6 million tweets relating to medical conditions gathered in the first six months of 2014 and a crowd-sourced team of over one hundred people hoping to do useful analyses of that data.

The project faced two problems initially, which prevented useful progress: (1) the data was incomplete and (2) there was little understanding of the nature of the data. This report details an effort to solve these problems. It does not itself provide any breakthroughs but rather it serves as the baseline of data and analysis upon which further efforts can build.

Most people coming to a project to analyze social media face the same basic problems: they do not have a rich-enough data set to work with, they do not have a solid understanding of the nature of the dataset and they do not know the analytical tools available. This report details an effort to remedy these problems.

There were three steps:

1. Part I Data Acquisition and Management
   Beginning on page 5. The data available was very limited; it was voluminous, but fundamental features were missing. Furthermore, there were questions as to the most-appropriate form of data storage to support the analyses. This step solved these problems.

2. Part II Exploratory Data Analyses
   Beginning on page 9. All of the typical analytical tools were applied to the expanded dataset:

   - Time Series
   - Geographical Plotting
   - Online analysis of real-time Twitter data
   - Sentiment Analysis
   - Textual Analysis
   - Network Analysis

3. Part III Research into Similar Projects
   Beginning on page 35. Other projects of a similar nature to this one have been done. References have been provided to allow members of this project to benefit from the work of others.

Part I
Data Acquisition and Management

2 Collect Twitter Data

The first step was to add all the Twitter data to the files provided by the project.

The project was provided with 896 csv files by Topsy, a Twitter aggregator [Topsy, 2014], containing well over 6 million tweets [Google Drive, 2014]. The files contain tweets concerning a wide range of medical conditions for the first six-months of 2014.

However the data included only the text of the tweet, its originating user and a score calculated by Topsy. While this might be sufficient for a rudimentary textual analysis, the other data provided by Twitter is clearly of value for more extensive analyses: even simple analyses such as filtering by retweet count, time-series studies or plotting geographic incidence are impossible with the data provided.

My GitHub repo for this project [Fisher, 2014b] contains a python program that requests the full json from Twitter for each tweet by its id. I transferred the project files to Amazon Web Service's EC2 service and ran the python program against them all, producing files that anyone can download for their own use from S3 [Fisher, 2014a]. See the appendix on page 39 for details of the files.

2016 note: I've created a web page on Amazon Web Services that makes all the files referenced in this paper much easier to access for your own work: http://healthcare-twitter-analysis.com.s3-website-us-west-1.amazonaws.com/

3 CSV vs. JSON

Initially, I focused on creating csv files with this data, and the programs to do so are still on the repo, but after studying Twitter analysis I became convinced that json was more appropriate:

1. json is what Twitter itself uses and all of the Twitter documentation reflects that.
2. Every book and paper I have read and every Twitter-analytic program refers to the Twitter data in its json form
3. The primary reason json is preferred is that it permits structures that are more complicated than just rows and columns; for example if one tweet has two hashtags and another has four, the dictionary-like structure of json can easily support this, expanding or contracting as the need arises. XML is another structure that provides this capability but it is considerably more of a pain to use than json.
4. While MongoDB [MongoDB, 2014] does support csv files, including a utility for csv loading, MongoDB's native document structure is json and an imported csv file is converted to json for storage in the database.

5. json also reflects Python's native data structures of lists and dictionaries; json, Python and MongoDB are often used together for this reason.

4 Text, SQL, NoSQL

On the assumption that this project unearths some really useful analytics that can help medical science, it will need to address the question of the best way to store the data. We're using text files at the moment which are easy to use but they get clumsy quickly.

Through 2010 Twitter used MySQL for its data storage. ([highscalability.com, 2011, Wired, 2014, Quora, 2012]). Subsequent volume growth and the need to serve data from many locations worldwide prompted Twitter to build its own NoSQL database [Twitter, 2014a, Computerworld, 2010].

Initially I thought we could consider MySQL since Twitter itself had used it but the way it was used was to store the json in a single long text field and to use UDFs to parse it. This is clearly inferior to using MongoDB which does essentially the same thing but is specifically built for indexed json queries.

MongoDB, like all NoSQL datastores, requires map/reduce to perform join operations. Therefore we must load into MongoDB records that have all the data we need to process a single tweet rather than compositing data from several tables as relational databases do. If we find useful data in addition to the Twitter data and the supplemental data from this project (ontologies, perhaps), it will have to be incorporated into the json record.

See the sample program listing in Appendix D on page 49 for an example of loading a MongoDB database with the Twitter json and searching it using Python. It also includes examples of using MongoDB's geo-spatial indexes which would allow you to find, for example, the 100 tweets nearest to a given location on a given day, containing a particular hashtag.

2016 note: AWS provides a managed service for NoSQL databases called DynamoDB. I have come to the conclusion that using AWS-provided services is far more efficient than DIY and I recommend you have a look.

5 Supplemental Data

5.1 Twitter & Geo Data

The work detailed in this paper resulted in the data listed below being added to the Topsy csv files, each record being in json format exactly as produced originally by Twitter with fields added with the new data.

See appendix A on page 37 for the details of the new fields in json format; see page 40 for a link to a file with 4 million tweets containing these fields; see page 46 for examples of using this file with R, Python and MongoDB.

See Appendix E for notes on geo data.

- All the Twitter data
- Unix timestamp
- Latitude & longitude
- Country name & ISO2 country code
- City
- For country code "US"
  - Zipcode
  - Telephone area code
  - Square miles inside the zipcode
  - 2010 Census population of the zipcode
  - County & FIPS code
  - State name & USPS abbreviation

5.2 Ontologies

Tim Cook [Cook, 2014] has begun work on adding ontology data from BioPortal [BioPortal, 2014][1]

[1]In this context an ontology is a hierarchy such as wine, red & white wine, varietals, etc. in the areas of symptoms, diagnoses and drug applications. It seems reasonable to suppose that such categorizations could expand the scope of the analytical possibilities but at the time of this writing this data has not been incorporated. It might be a useful project to do so if someone could figure out an application.

6 Starting Place for Further Analysis

The file `HTA_noduplicates.json` is the final product of the reverse geo-tagging process with all duplicate records removed (a surprising number of duplicates was found). This file can be downloaded in gzip format from the link on page 40.

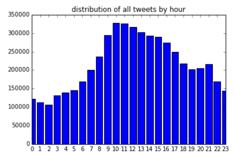
Appendix D on page 46 contains examples of processing this file with R, Python and MongoDB and constitutes the best starting place for anyone looking to analyze this data.

Appendix A on page 37 shows the new json fields added to the data and contains a link to Twitter's API, which shows the format of the json of every tweet. No analysis will proceed very far without at least a summary understanding of these fields.

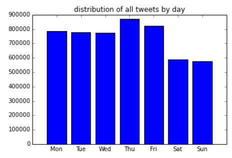
The following sections show examples of a number of basic analyses that are commonly done with Twitter data. My hope is that, with the work of creating a clean and (moderately) comprehensive dataset out of the way, these examples will trigger ideas about correlations or visualizations that will help make this data useful in some way to the medical community.

Part II
Exploratory Data Analysis

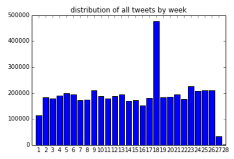
7 Time Series Analyses



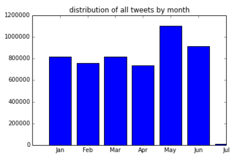
(a) By Hour of the Day



(b) By Day of the Week

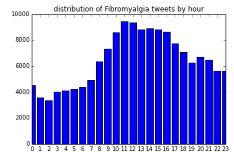


(c) By Week

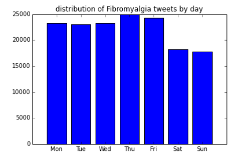


(d) By Month

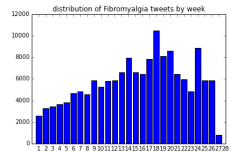
Figure 1: Distribution through time of **all tweets** in the database. Week 18 is unusually well represented.



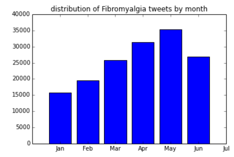
(a) By Hour of the Day



(b) By Day of the Week



(c) By Week



(d) By Month

Figure 2: Distribution through time of **Fibromyalgia**. Perhaps we can see a seasonal pattern beginning to emerge; we would need data for the rest of the year to tell ... two years' worth would be better still. (Note: I asked a practicing Rheumatologist about this and she said that she was not aware of any seasonality in Fibromyalgia flareups.)

8 Plotting Data on a Map

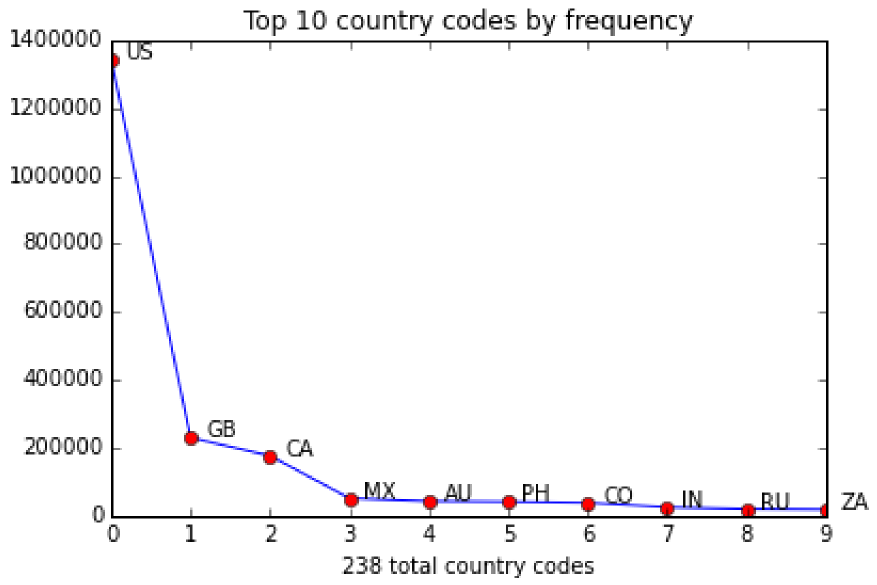


Figure 3: **Top 10 Country Codes in the Dataset**

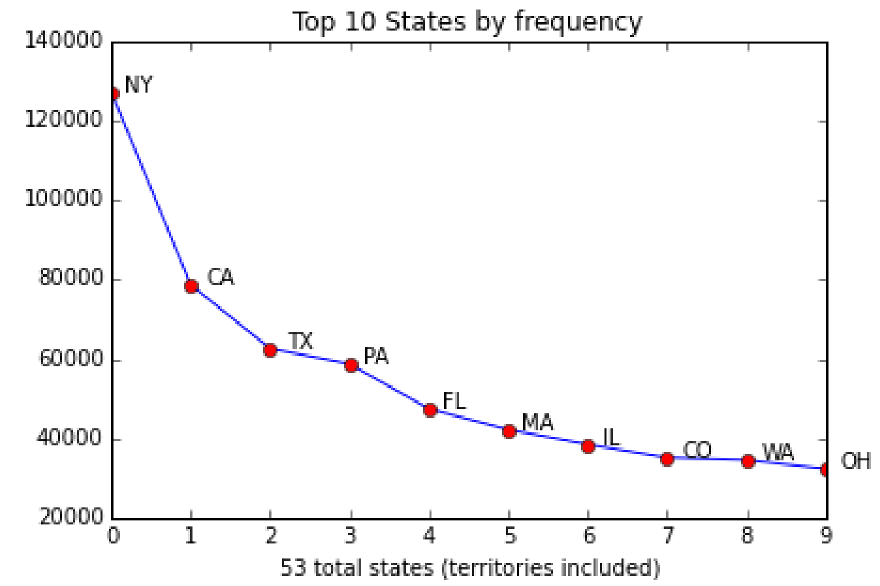


Figure 4: **Top 10 States in the Dataset**

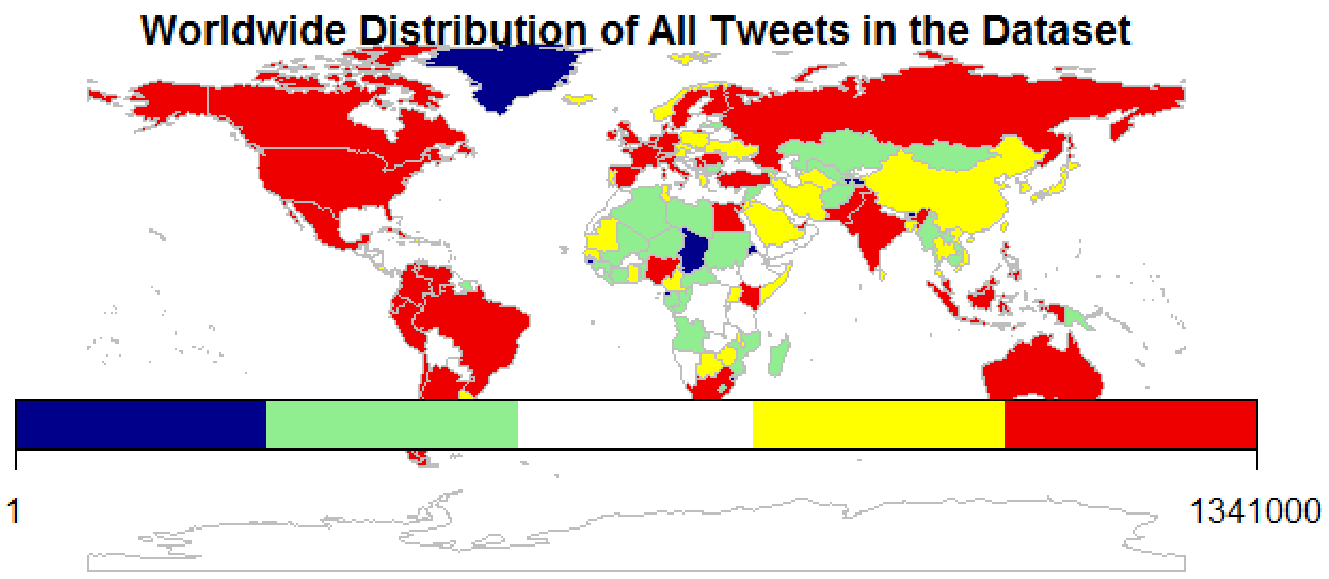


Figure 5: **Red: the heaviest users, the top 20%; Yellow: the runners up**

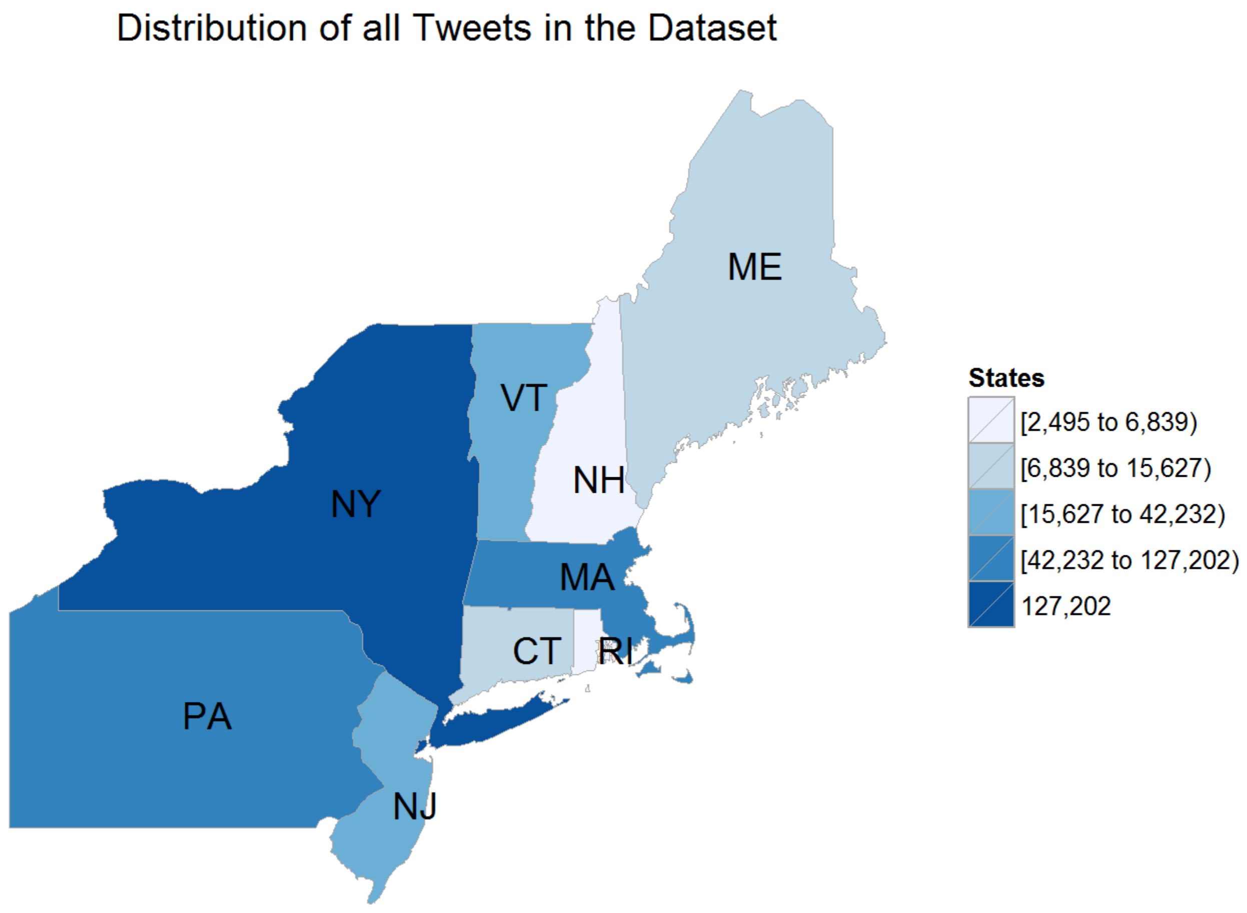


Figure 6: **Distribution by US States in the North East**

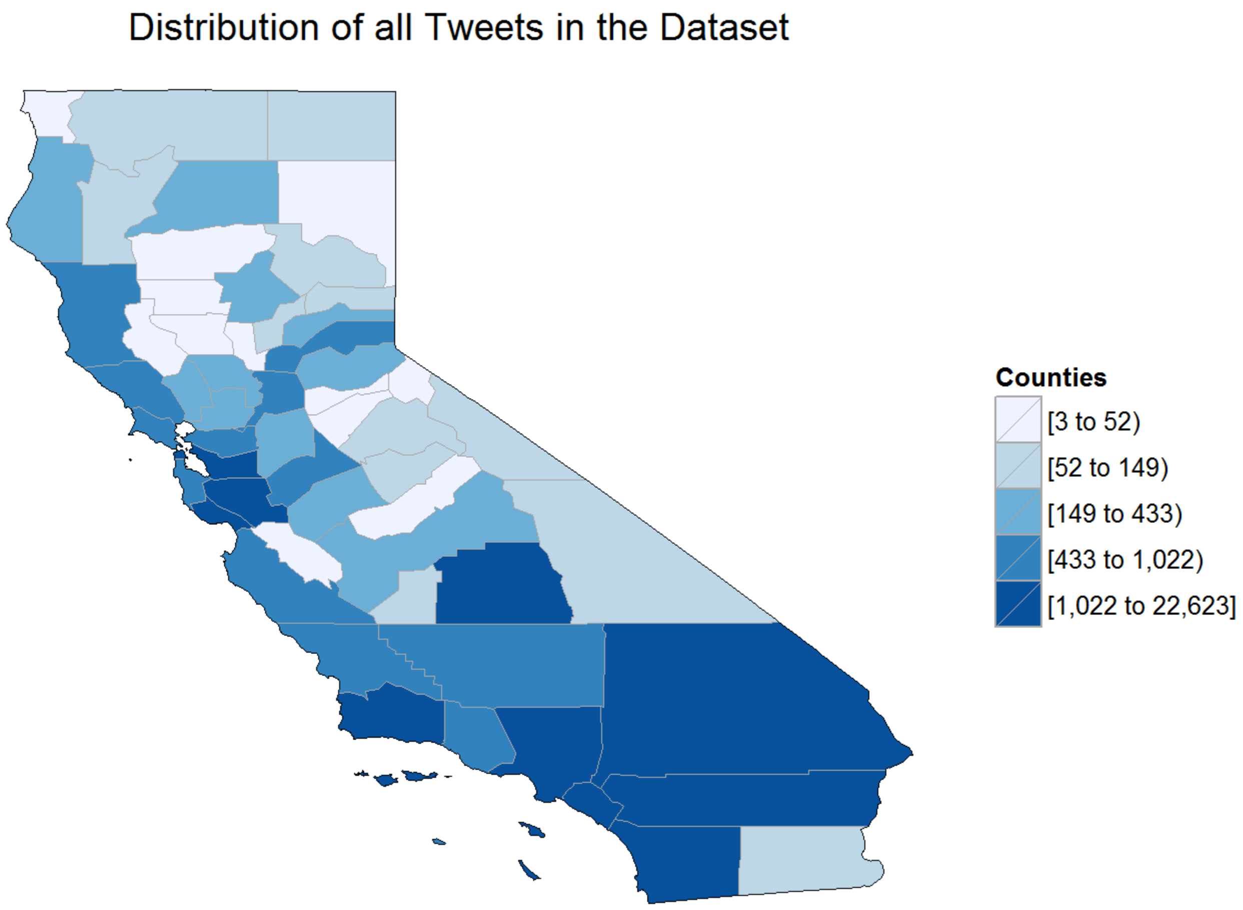


Figure 7: **Distribution by County in California**

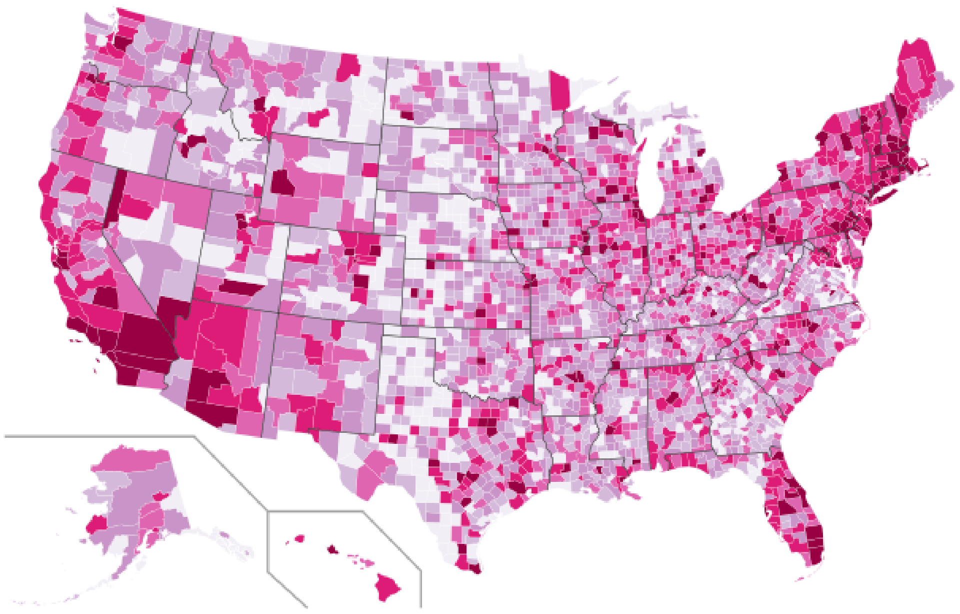


Figure 8: **Distribution by all US Counties**

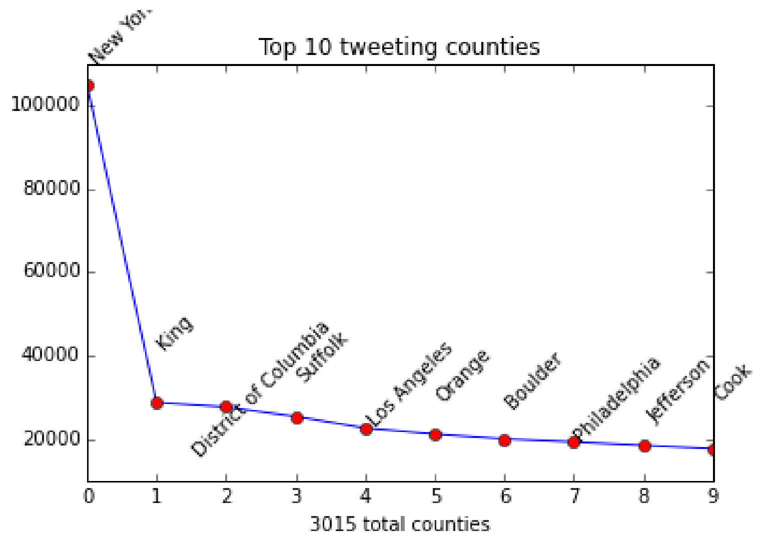


Figure 9: **Top 10 tweeting Counties in the US**

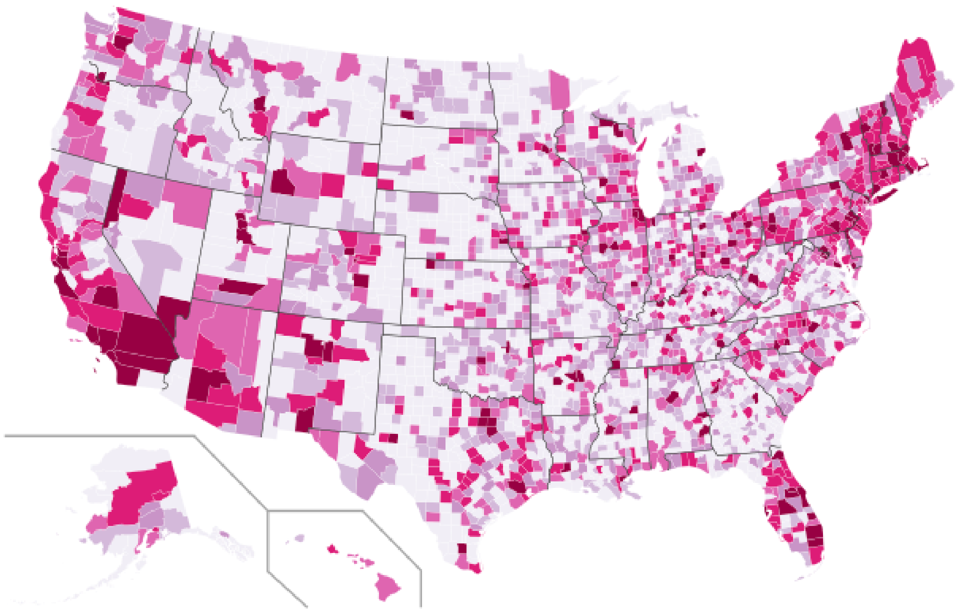


Figure 10: **Distribution by US Counties of tweets mentioning depression.** Note: the fact that big population centers have large numbers of people mentioning *anything* may not be significant; you can use the 2010 census data (included in the supplemental json data of every record) to derive measures by population density or per capita which may be more meaningful.

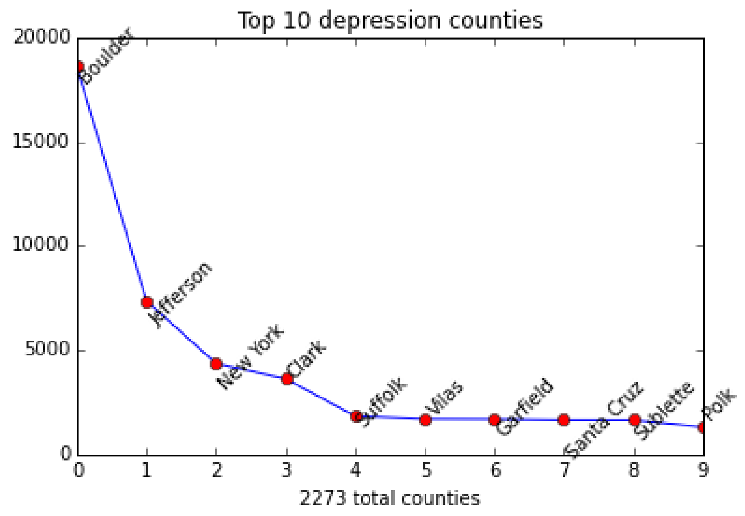


Figure 11: **Top 10 US Counties mentioning depression**

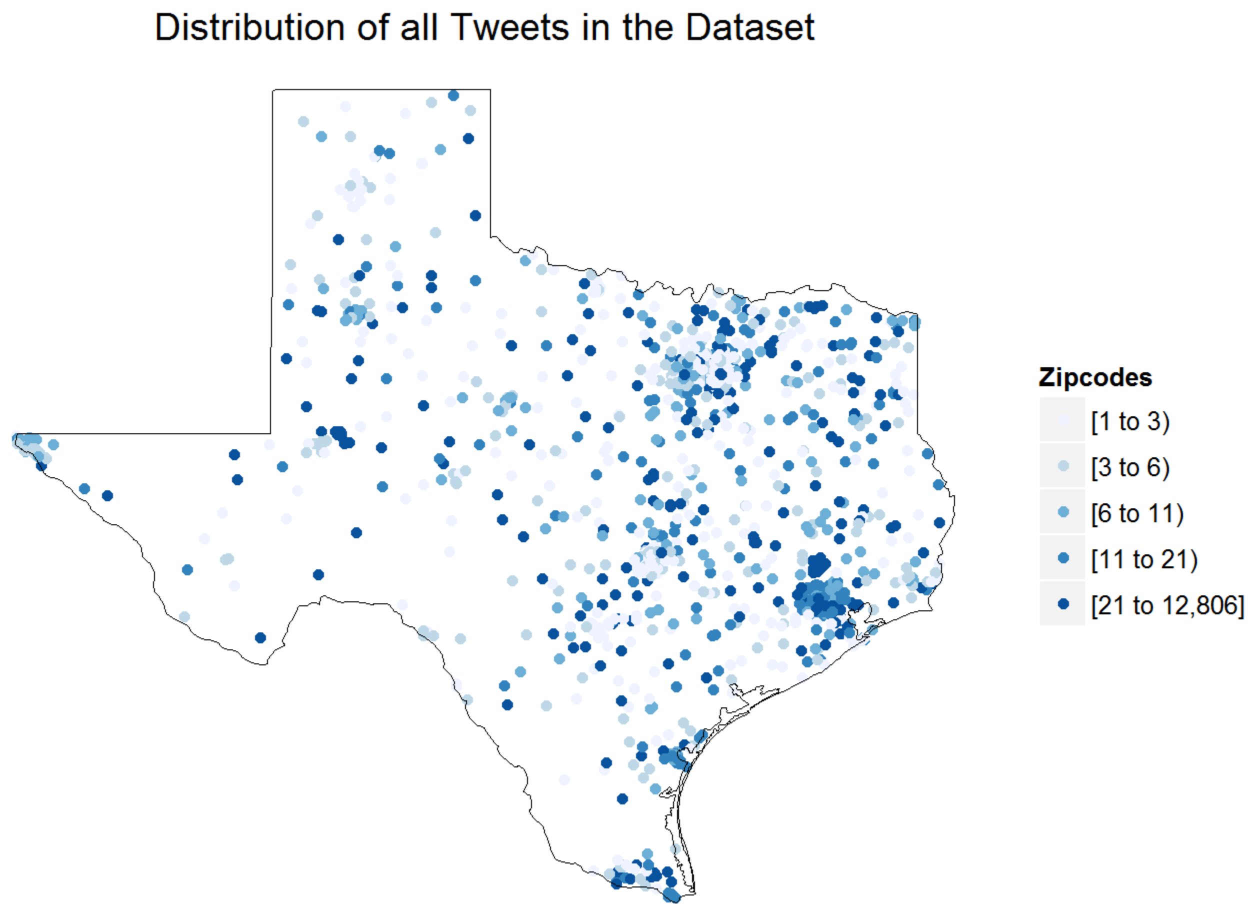


Figure 12: **Distribution by zipcode in Texas**

9 Online Twitter Access

9.1 Online with Python

The main section of the repo contains `Instructions for python.pdf` which provides instructions for setting up Python, IPython and installing the prerequisites for online Twitter access.

The `code` folder of the repo contains an IPython notebook `Online Twitter Basics.ipynb` that walks through the process of making online-queries of Twitter and doing simple analyses of the responses. From the notebook you can combine the static project data with real-time queries.

9.2 Online with R

The main section of the repo contains `Instructions for r.pdf` which will get you set up for online Twitter access from RStudio, which is where I did most of the analyses in this document, some of it using the static project data, some it doing real-time Twitter queries.

9.3 Analyzing the Static Project Data

Appendix D on page 46 contains sample programs for processing the static json project data.

2016 note: As long as the data is on AWS, why not do the analyses entirely on AWS? There are numerous websites instructing you on how to get Jupyter (IPython) Notebooks and RStudio running on an AWS EC2 instance; I have done this myself and I think it's a superb way to go.

10 Sentiment Analysis

10.1 Summary of Sentiment Analyses

Sentiment Analysis is a widely-used technique to infer the sentiment of a message from the words and phrases used, including emoticons. The Breen sentiment scoring system seems to be preferable to AFINN and I've run a few samples on subsets of the data. It is not clear to me that comparing sentiment between disease types is helpful; scoring individual tweets within a diagnosis may be more helpful in identifying candidates for further study.

10.2 Analysis of Breen, AFINN, Score

There are two sentiment measuring systems which popped up in my initial studies of the subject: Jeffrey Breen's [Breen, 2011b,Breen, 2011a] and AFINN [Nielsen, 2011]. In addition, the Topsy data includes a measure called score [Topsy, 2010].

I wondered how the two sentiment measures compared to each other and whether sentiment and score had any relation. Loading the data on Cancer, Cardiovascular and Digestive into R, I had a look:

	breen	afinn	score
min	-6.00	-10.00	6.02
mean	0.00	0.50	8.36
median	0.00	0.00	7.58
stdev	1.23	2.10	1.66
skew	0.00	0.65	1.05
npskew	0.00	0.24	0.47
kurtosis	0.85	2.76	-0.15
max	6.00	16.00	14.62

Table 1: **Statistical Comparison of Sentiments and Score**

10.3 Digging Deeper into Sentiment Measures

Gaston Sanchez wrote a series in 2012 about Twitter analysis [Sanchez, 2012]. His work provides an interesting overview of general summary analyses that people do on Twitter data and I have reproduced some of his work here, using R and the Breen sentiment scoring system [Breen, 2011b], with data from this project in four (randomly-chosen) categories :

1. Blood Disorders
2. Cancer
3. Cardiovascular Diseases
4. Digestive Disorders

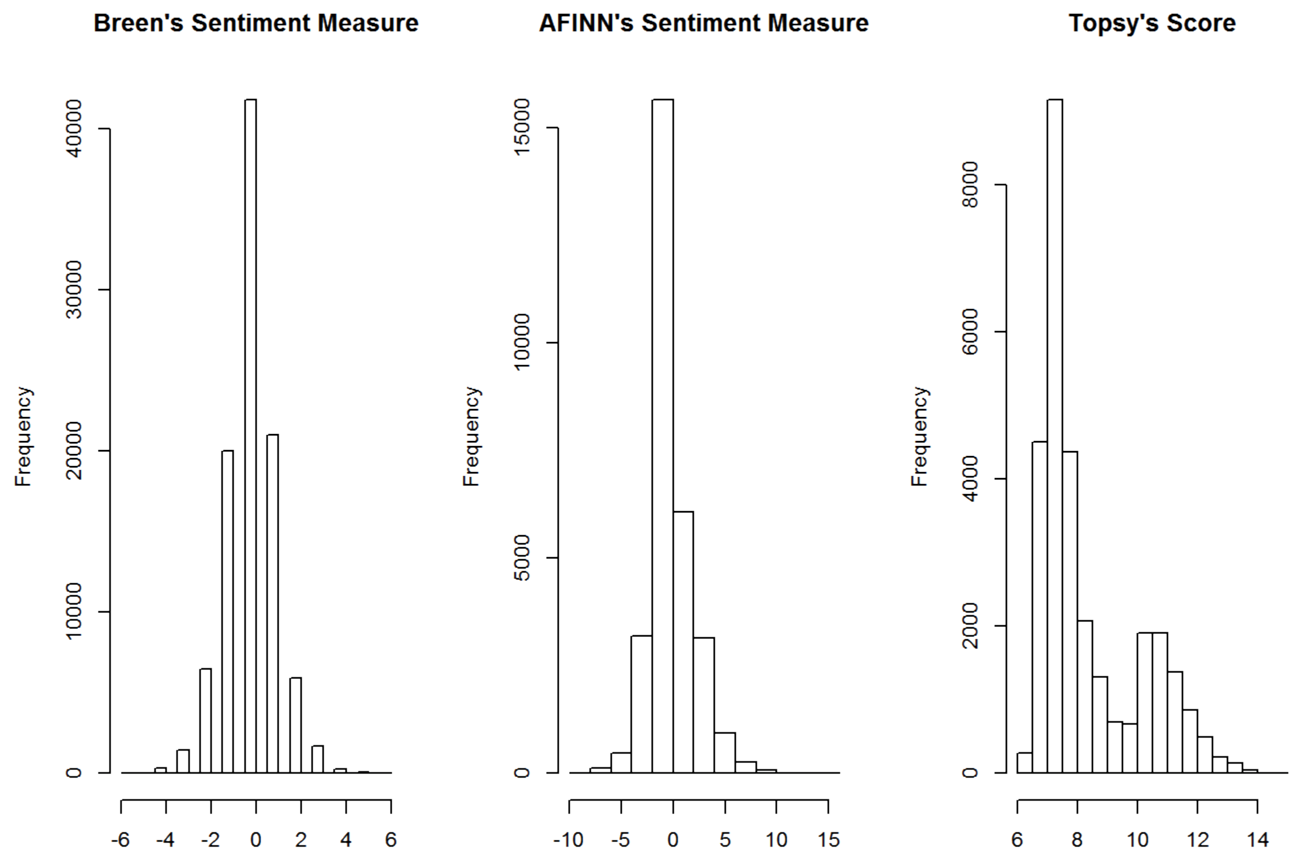


Figure 13: **Distribution of Sentiment Measures and Score** Breen and AFINN are more similar to each other than to score: both have a mean of nearly zero and both are symmetrical around it; but AFINN has a much greater variance and non-normal tail behavior. Score has more of a log or Poisson shape to its distribution, which is bimodal, and is clearly different from the two sentiment scores.

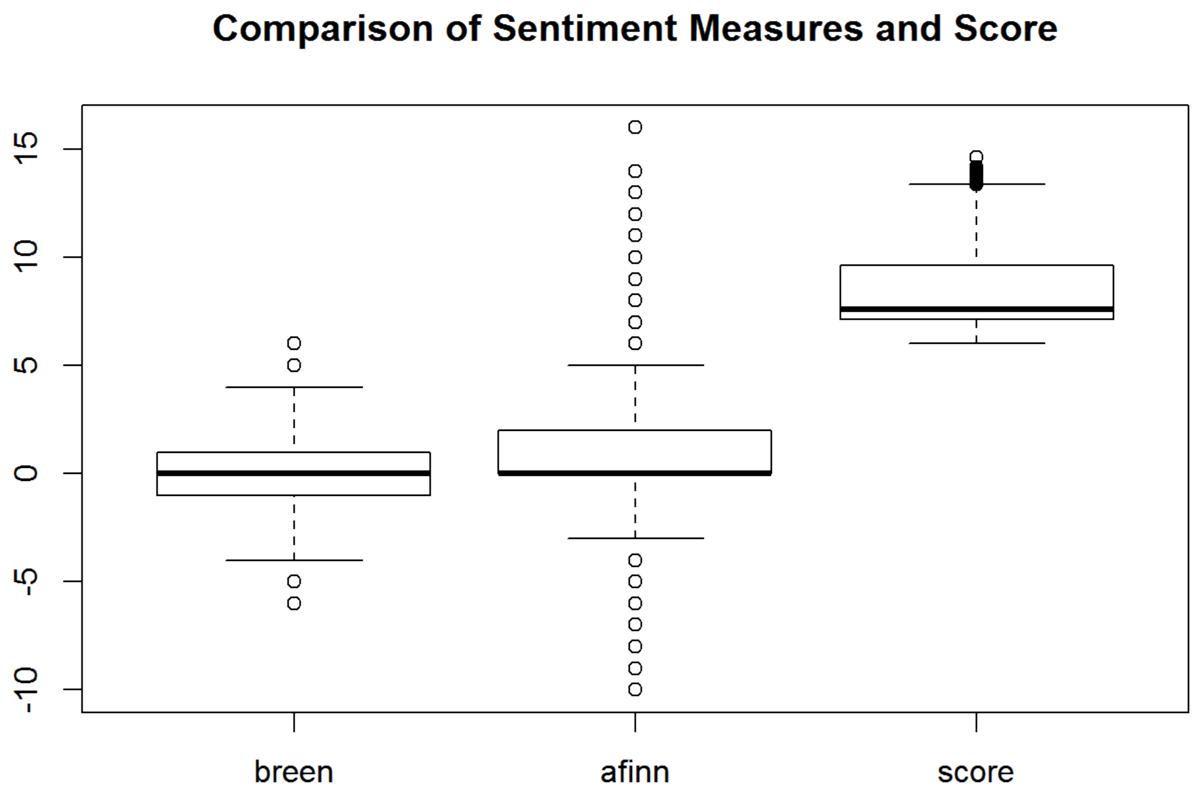


Figure 14: **Distribution of Sentiment Measures and Score** A box plot shows even more starkly the difference in the distributions of these three measures.
First It would seem that score is not created from or predicted by either sentiment measure.
Second The question arises as to which sentiment measure is preferable, if indeed either is adequate: AFINN has a much greater dispersion of its measures, which perhaps is to be expected when dealing with life-destroying diseases; on the other hand, Breen produces a more-nearly-normal distribution and by some accident of Providence, most naturally-occurring phenomena are normally distributed, perhaps including peoples' feelings.

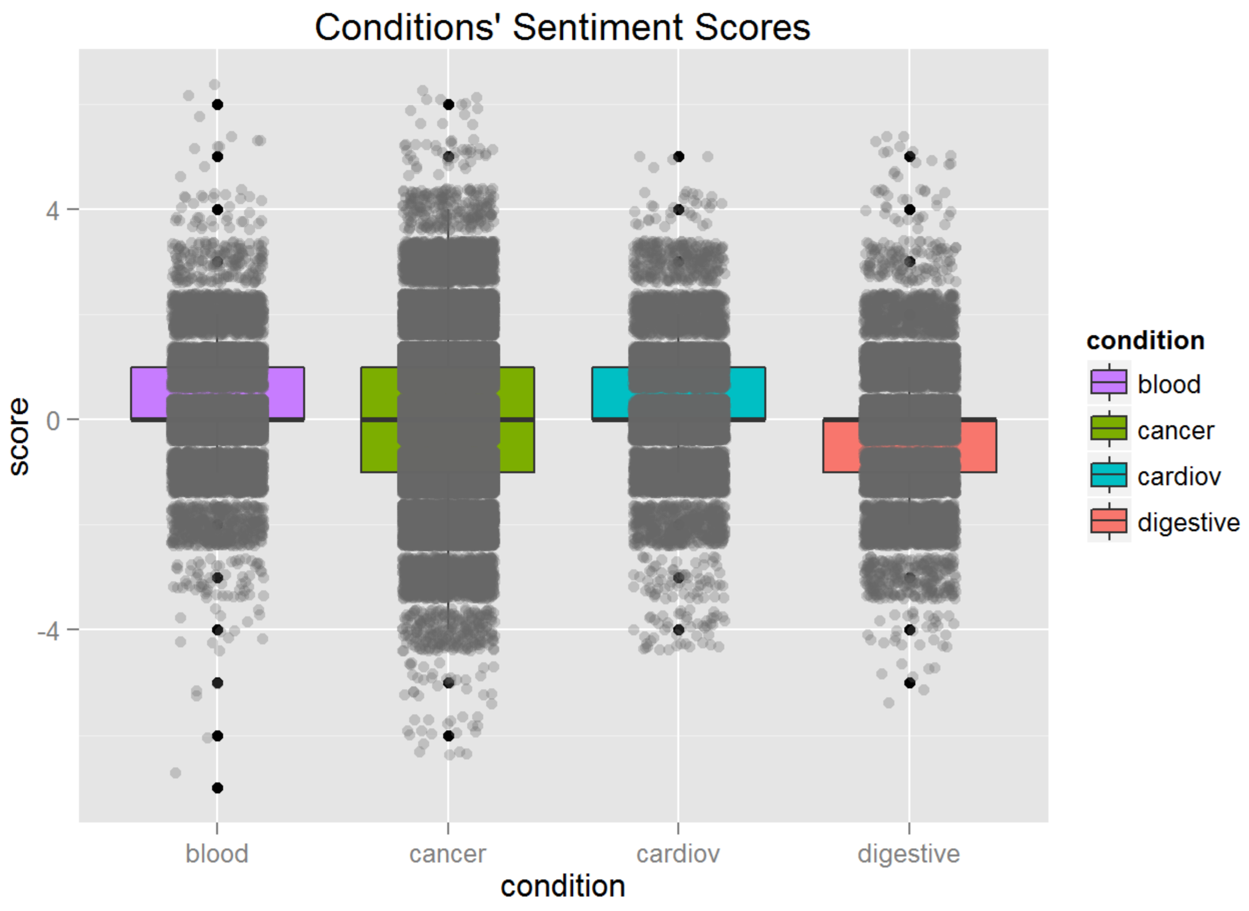


Figure 15: **Distribution of sentiment: Boxplots** The dark gray dots represent the individual data points, roughly 14,000 per condition. The boxes in color represent the inter-quartile distribution of the sentiment for each condition, with bold dots above and below representing outliers beyond the inter-quartile ranges.
They all have their median nearly at zero with a very wide dispersion in both the positive and negative direction. Blood and Cardiovascular disorders seem to be somewhat skewed toward positive overall sentiment while Digestive disorders are skewed toward the negative ranges.

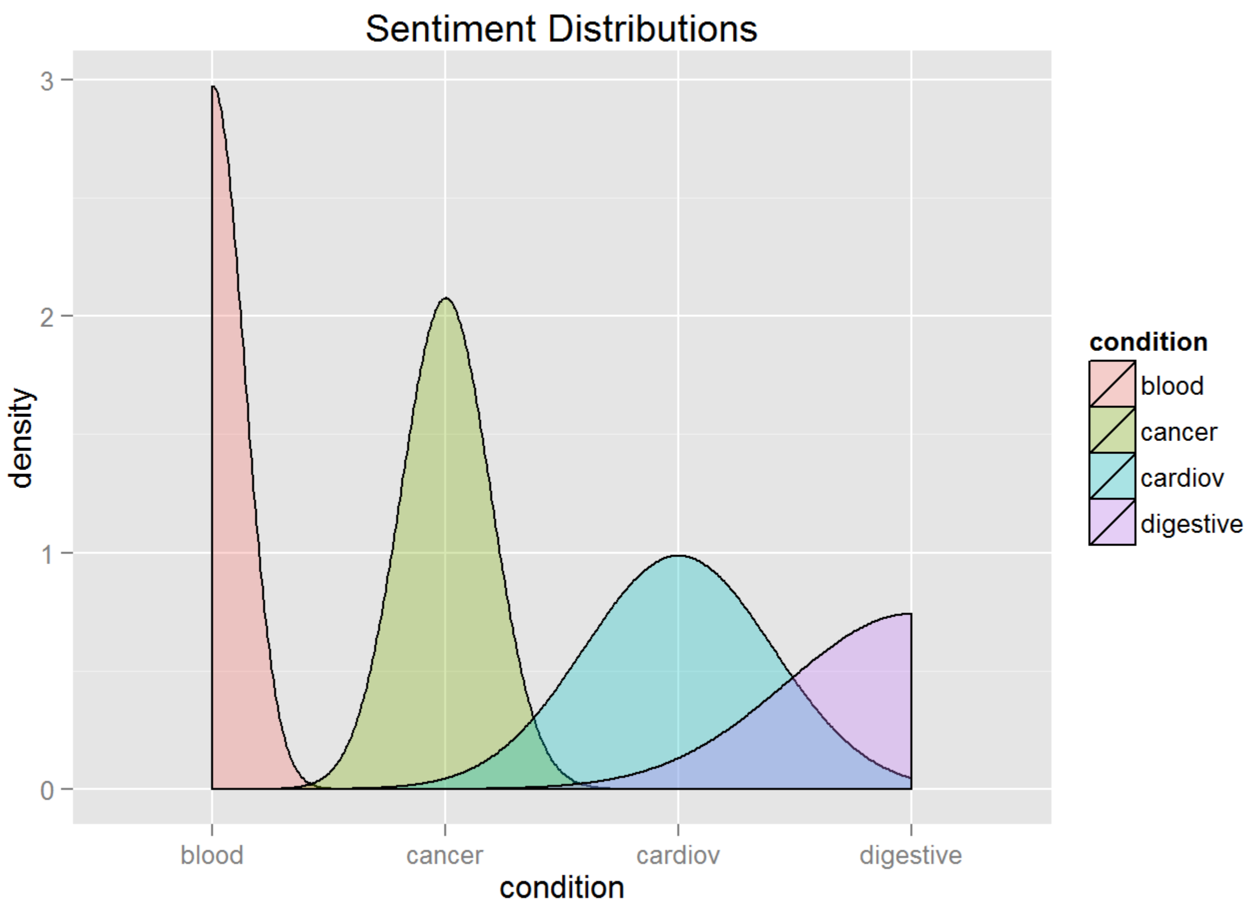


Figure 16: **Distribution of sentiment: Histograms** Another way to look at the distribution of sentiment is to show a smoothed histogram. For each condition, the vertical white line over the label is plotted over the average for that category and the plot shows the distribution around the mean although the left-tail of Blood and the right tail of Digestive are not plotted due to size constraints but they are roughly symmetric. In the study of sentiment measures in section 10.2 beginning on page 16, it was shown that the Breen sentiment measure is symmetric in general and the measures for these specific conditions reflect that.
Blood is in a tight range around its mean, while Digestive has the greatest dispersion.

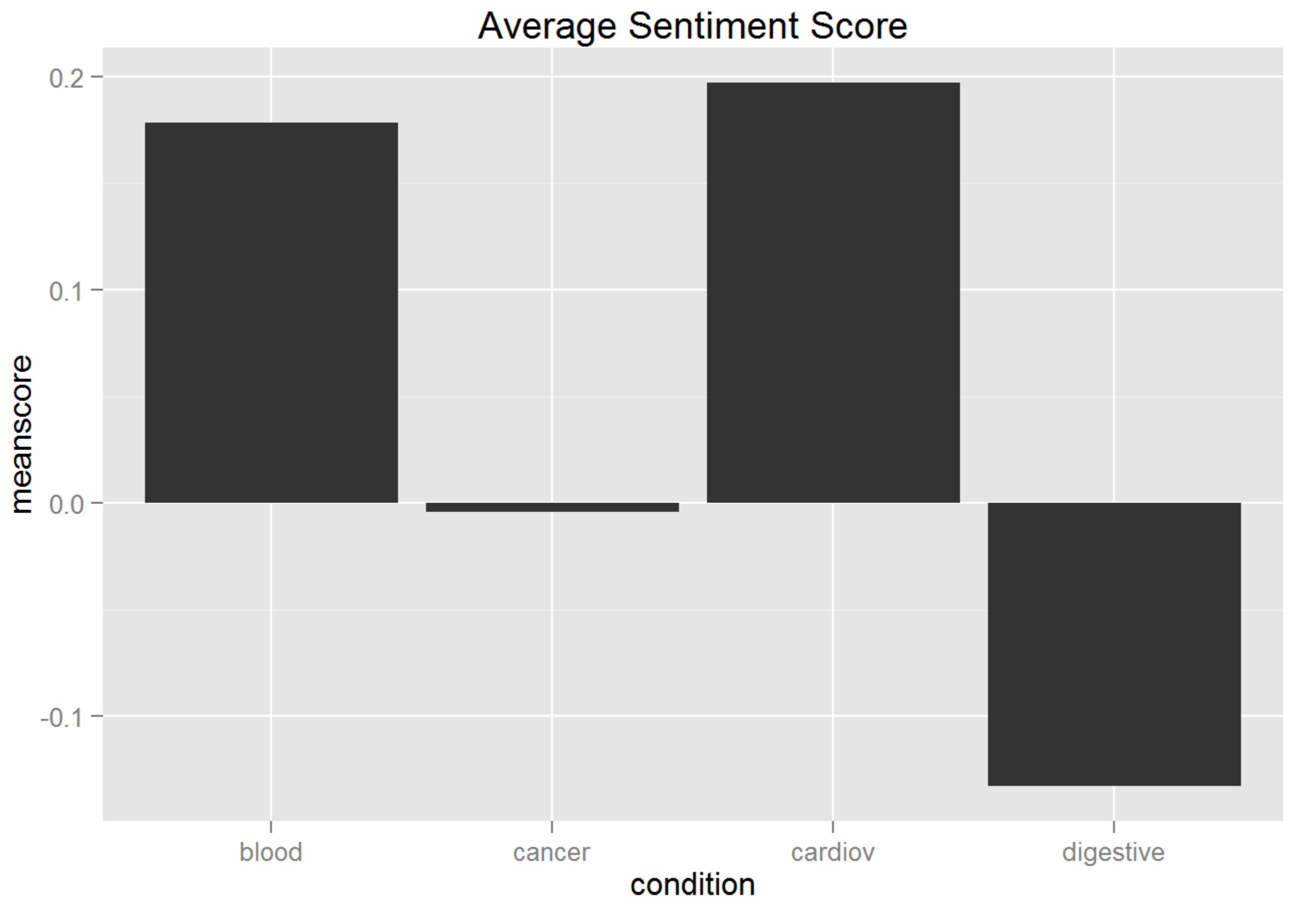


Figure 17: **Average Scores** The averages show us very starkly what we saw in the distributions: Digestive disorders seem to have by far the most negative effect on their suffers and/or those who tweet about them.

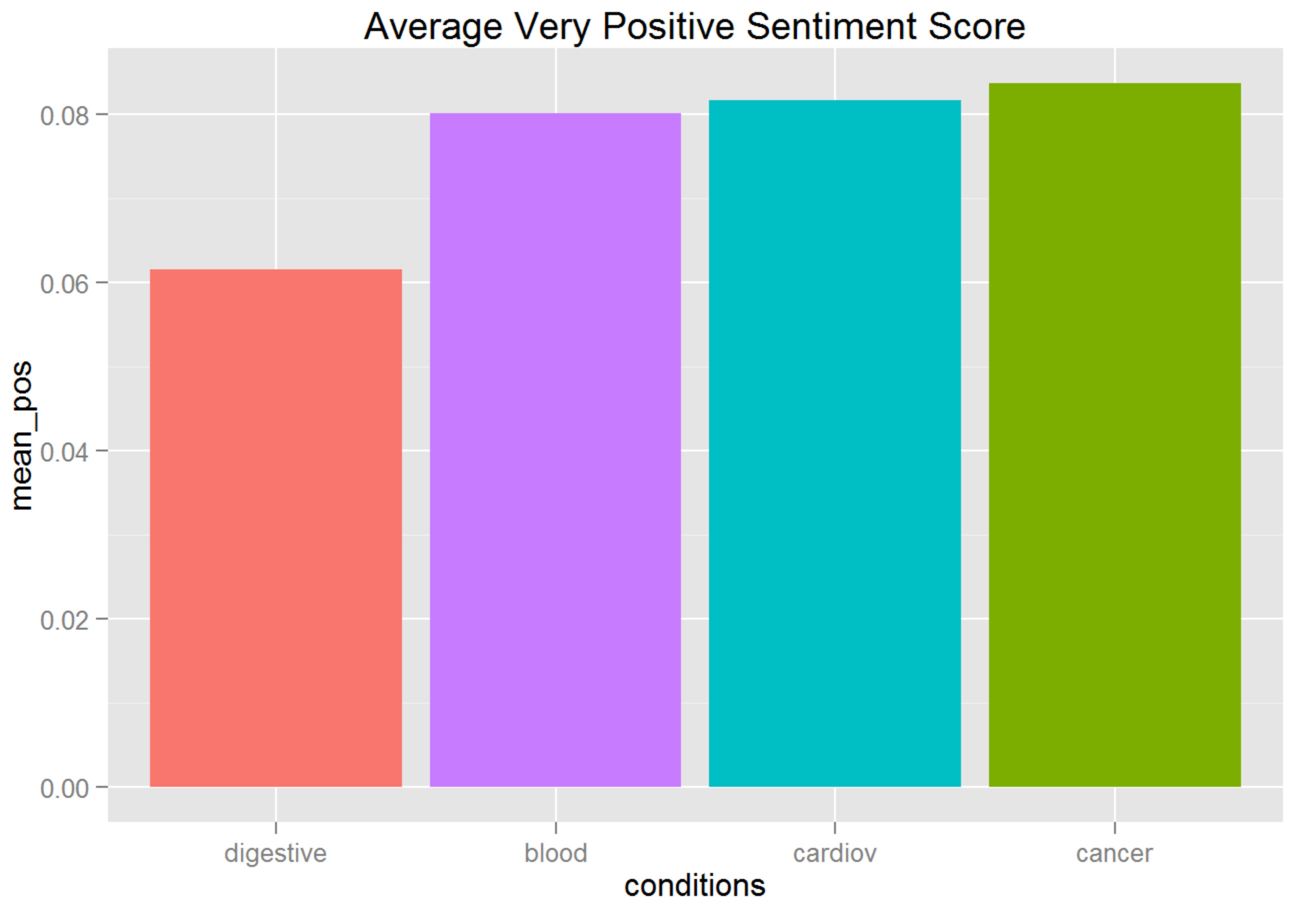


Figure 18: **Average Positive Scores** Looking at the mean scores for only those with a positive sentiment provides more reinforcement for what we have already seen: digestive disorders have a negative psychological effect to the extent of having the lowest mean positive scores.

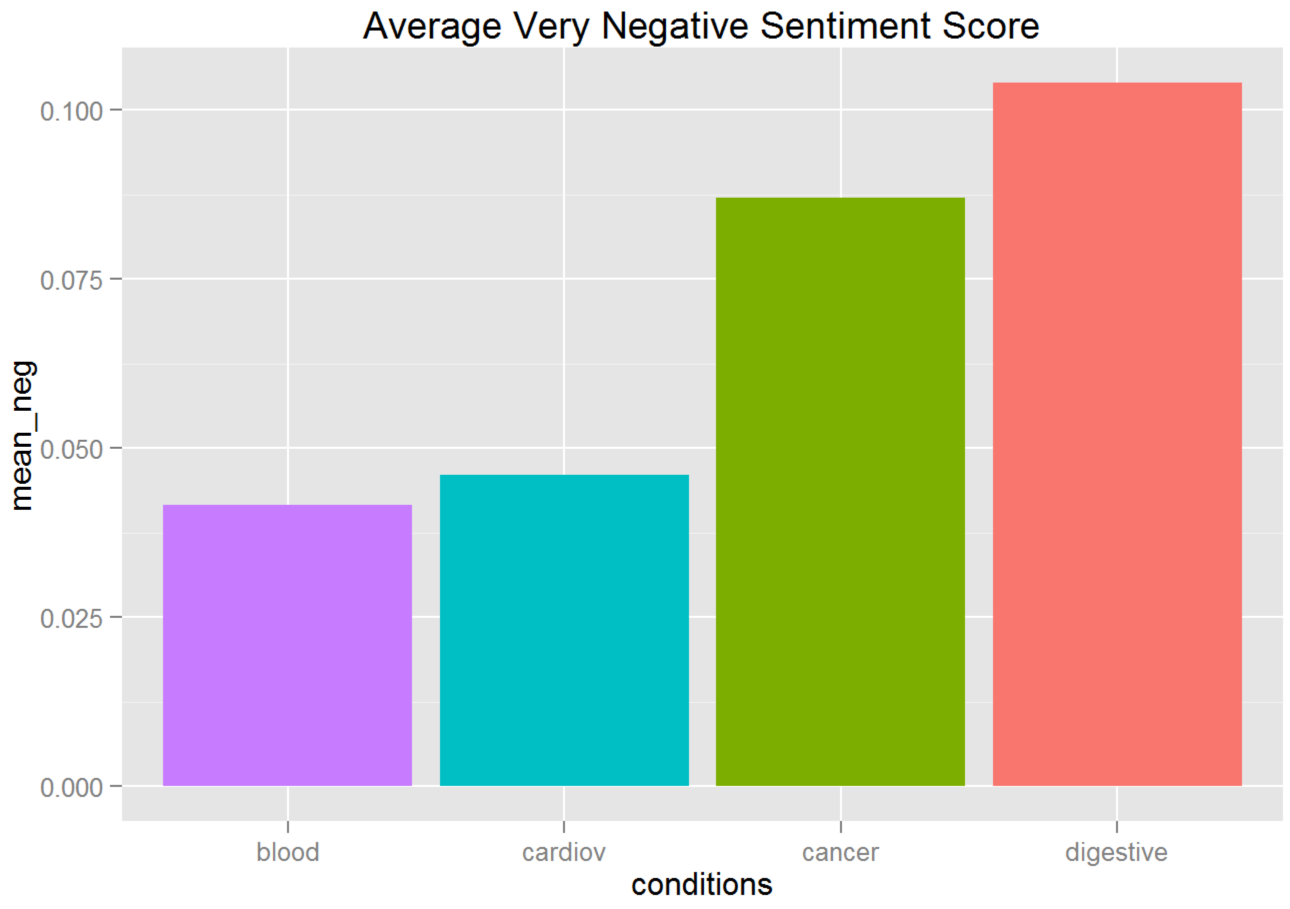


Figure 19: **Average Negative Scores** Looking at the mean scores for only those with a negative sentiment tells the same story: none are good, but of these four, tweets about Digestive Disorders show the greatest tendency toward negativity.

11 Analysis of Text

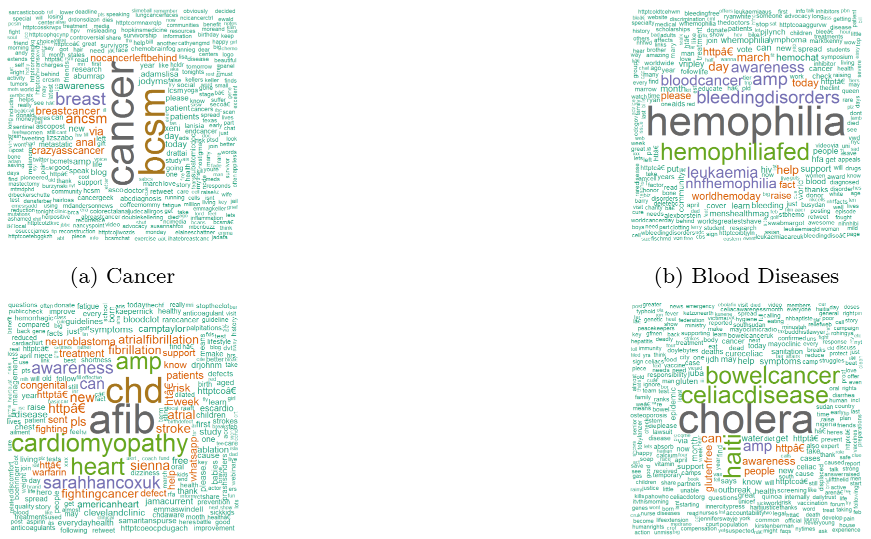
11.1 Summary

Textual Analysis is another popular analytical technique. By itself, it does not appear to add much value but it is possible that by including additional tags found outside the tweets themselves we would augment the results sufficiently to provide useful insights.

11.2 Word Clouds

Word Clouds are a very popular EDA technique for text and again with help from Gaston Sanchez [Sanchez, 2012] I have produced a sampling with R and datasets created using the technique described in section 2 starting on page 5.

The corpus was restricted to the first 10,000 tweets in the database for each condition and then further reduced to include only those that had been retweeted more than three times; without these filters the pictures were an incomprehensible mess.



(a) Cancer (b) Blood Diseases

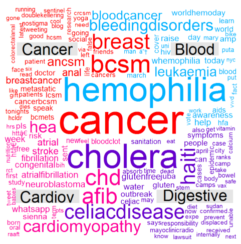
(c) Cardiovascular (d) Digestive

Figure 20: Simple Word Clouds for Four Medical Conditions

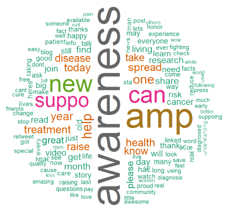
11.3 Word Frequency

The text field of a tweet has four kinds of 'tokens':

- Hashtags, beginning with '#', indicating a topic which is propagated across other social media to which the user belongs.
- User Mentions, beginning with '@', indicating a message to/about about a particular user
- URLs, links to other pages or media



(a) Comparative



(b) Commonality

Figure 21: **Comparative Word Clouds** show the words specific to the individual conditions. **Commonality Word Clouds** show the words that tweets about the four conditions have in common.

- Words, including some emoticons

I have parsed every text field in the database into these four token types, removing stopwords and nuisance strings such as 'rt' in the case of words, looking at the various frequencies of tokens:

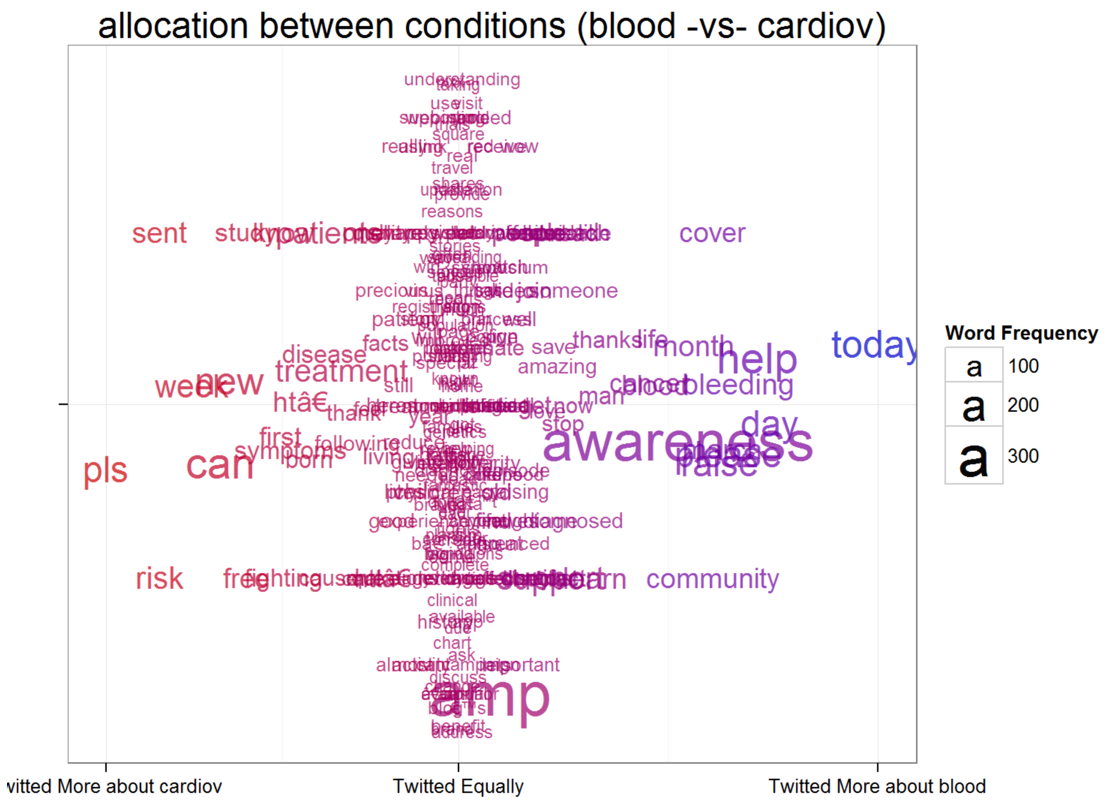
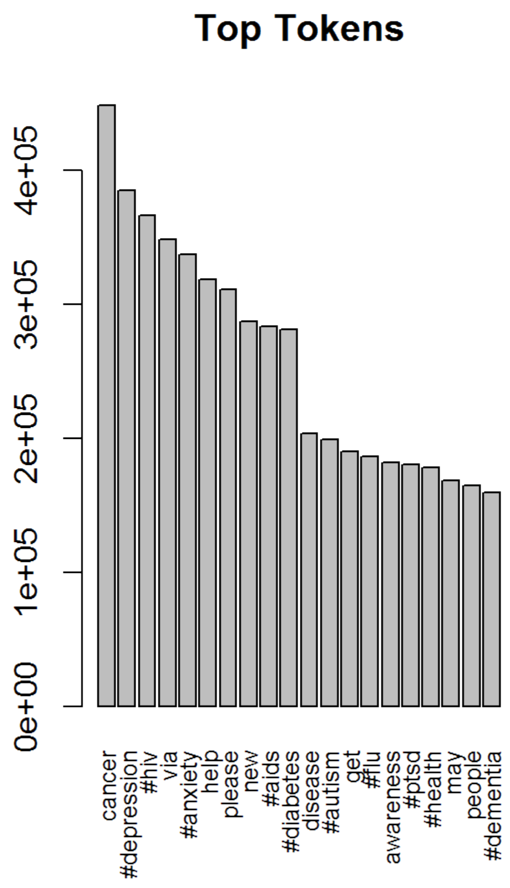


Figure 22: **Conway Comparative Word Cloud of Two Medical Conditions** Comparative word clouds compare all categories together. Conway word clouds show how two categories allocate words between them.

11.3.1 Overall Frequencies



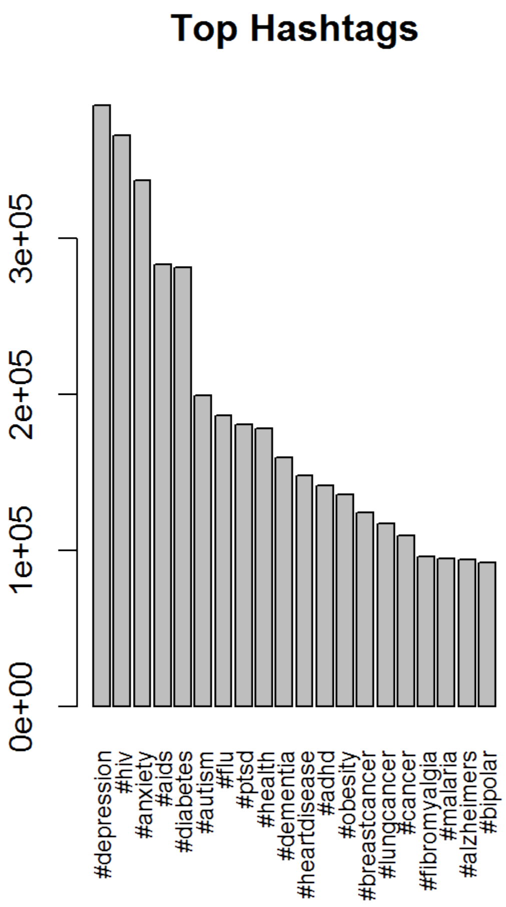


Figure 23: **Most Common Tokens Overall and Most Common Hashtags** The following hashtags are among the top hashtags mentioned in the entire dataset but are not on the list provided by the project:

- #health
- #cancer
- #mentalhealth
- #fibro
- #love
- #pain
- #awareness
- #veterans
- #asd
- #spoonie
- #disability
- #sex
- #weightloss
- #stress
- #glutenfree
- #advice
- #dating

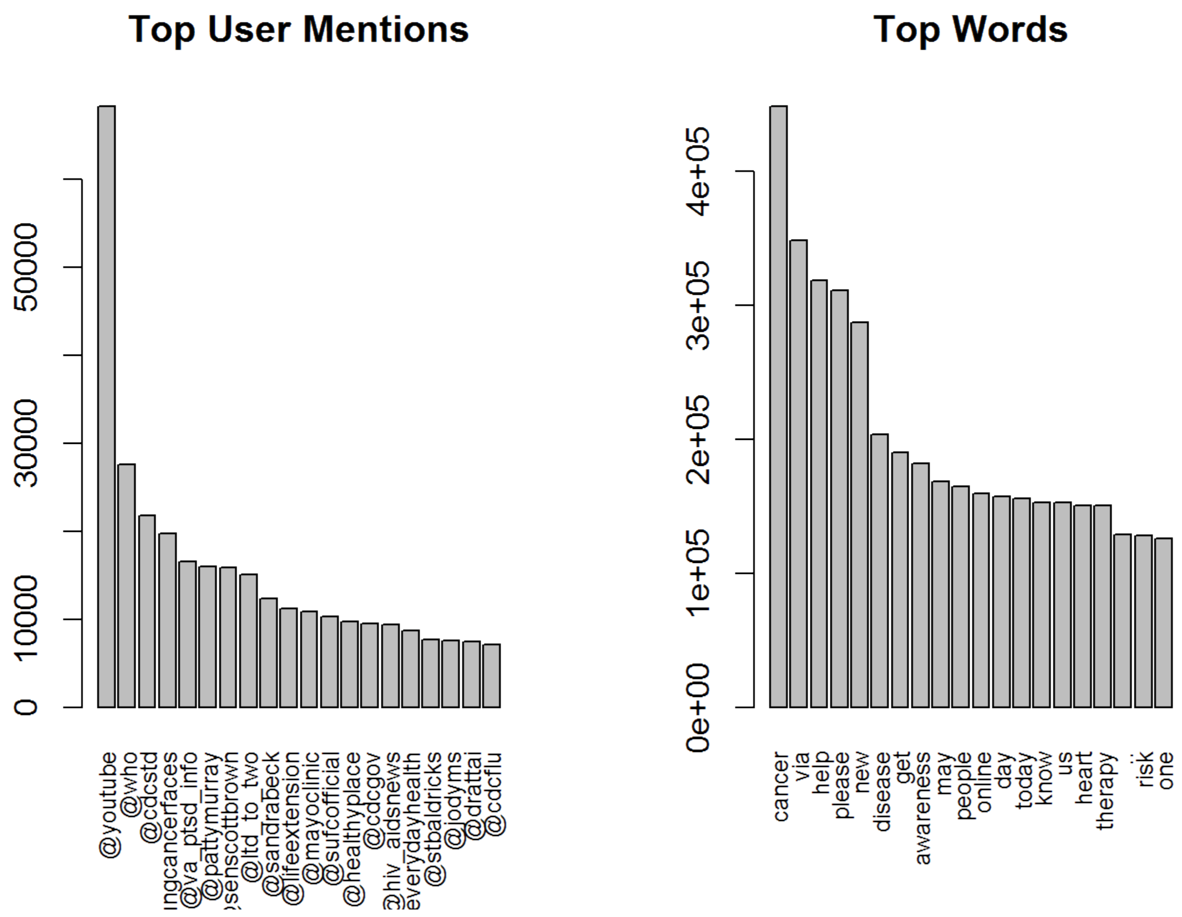


Figure 24: **Most Common Users Mentioned and Most Common Words Used**

Screen Name	Twitter Description
@youtube	Tweets on news, music and trends from all your favorite channels.
@who	Official Twitter account of the World Health Organization
@cdcstd	Helping people to be safer and healthier by the prevention of STDs
@lungcancerfaces	Faces of Lung Cancer
@va_ptsd_info	National Center for PTSD
@pattymurray	Senator Patty Murray
@senscottbrown	Scott P. Brown
@ltd_to_two	Multiple Sclerosis (PRMS), Fibro may have limited me but it can't destroy me.
@sandrabeck	Sandra Beck #TalkRadio Host #divorce #death #illness #recovery #faith #spiritu
@lifeextension	The latest research on health, wellness, nutrition, & aging.
@mayoclinic	The Mayo Clinic
@sufcofficial	Official Twitter site of Scunthorpe United football club.
@healthyplace	Trusted information on psychological disorders and treatments,
@cdcgov	Centers for Disease Control & Prevention
@hiv_aidsnews	News and developments in the global fight against HIV and AIDS.
@everydayhealth	Powerful weight-loss tools, expert advice & health news and information.
@stbaldricks	Charity funding the world's most promising research to #ConquerKidsCancer.
@jodyms	Writer, blogger. Optimist. Cancer Advocate.
@drattai	Breast Surgeon, President-Elect of @ASBrS
@cdcflu	Flu-related updates from the Centers for Disease Control & Prevention.
@icombat_stress	Motivational Mentor. Hope. Help. Healing. You CAN Turn Your Life Around.
@pozmagazine	The premier HIV/AIDS advocacy
@mndassoc	The Motor Neurone Disease Association.
@clevelandclinic	The Cleveland Clinic
@alldiabetesnews	The Most Comprehensive Diabetes News Aggregator on the Web.

Table 2: **Top Users Mentioned** One current and one ex Senator make the top users mentioned? A soccer club? . . . must have been gathered during the World Cup.

11.3.2 Co-Occurrence With Top Hashtags

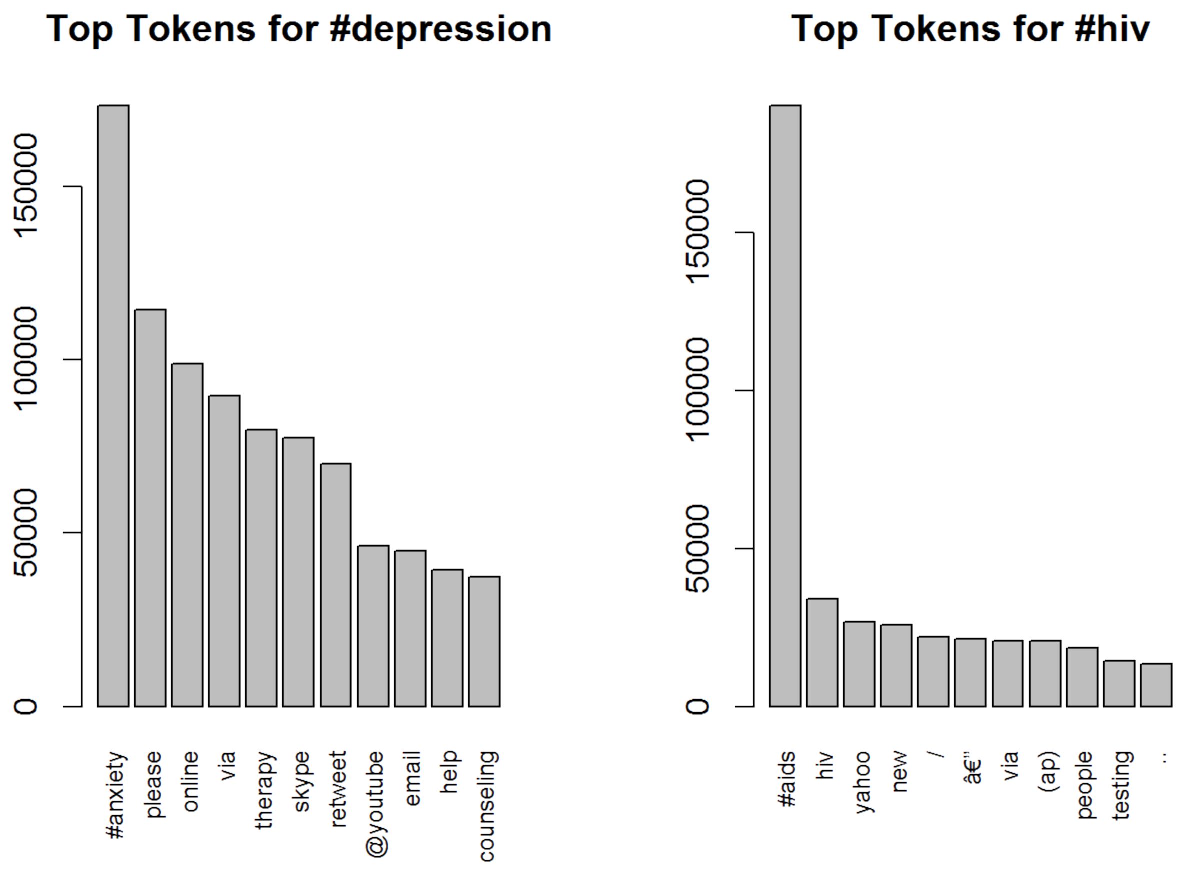


Figure 25: **Tokens Most-Commonly Co-Occurring With Two Top Hashtags**

11.4 Latent Dirichlet Allocation

I loaded 40,000 tweets of the Blood category into a matrix in R and asked it to tell me the topics; it did a pretty good job: it said there were three:

- sepsis
- myeloma
- hemophilia

11.5 1-, 2-, and 3-Grams for each Hashtag

The following are samples of three csv files in the repo that contain the most-common 1-, 2- and 3-grams associated with each of the hashtags for this project:

hashtag	1-gram	count
rettsyndrome	help	724
influenza	flu	5826
caudaequina	syndrome	20
schizofrenie	van	7
bedwetting	child	95
epilepsy	help	3913
dysautonomia	sharing	915
ppd	postpartum	1089
eds	awareness	1402
sarcoidosis	via	406
trichotillomania	hair	362
afib	atrial	1036
gallbladder	pain	599
testicularcancer	awareness	861
hernia	surgery	123

hashtag	2-gram	count
rettsyndrome	awareness for	250
influenza	out stories	990
caudaequina	please watch	7
bedwetting	your child	59
epilepsy	check out	878
dysautonomia	for sharing	843
ppd	postpartum depression	508
eds	ehlers-danlos syndrome	562
sarcoidosis	news daily	334
trichotillomania	check out	106
afib	atrial fibrillation	931
gallbladder	can cause	274
testicularcancer	to check	225
hernia	detailed general	30

hashtag	3-gram	count
rettsyndrome	$ awareness for	220
influenza	is out stories	990
caudaequina	please watch share	7
schizofrenie	dialoog finse blijkt	3
bedwetting	fitted mattress cover	23
epilepsy	thanks for the	649
dysautonomia	thanks for sharing	760
ppd	should feel ashamed	165
eds	info 085251378519 atau	352
sarcoidosis	news daily review	330
trichotillomania	support this eye-opening	90
afib	with atrial fibrillation	156
gallbladder	can cause severe	273
testicularcancer	about going through	156
hernia	general surgery videos	30
incontinence	disposable pads shaped	211

12 Network Analyses

NodeXL [CodePlex, 2014] is a software package in the form of an Excel template that provides network analysis and visualization. The Python package `networkx` provides a complete programmer's interface for network development and analysis. Neo4j [Neo4j.org, 2014] is a database that can hold RDF triples and supports the SPARQL query language; the queries can also be implemented in SQL and if the relationships are derived from multiple sources SQL may be a better choice. With the Twitter json, especially if combined with external 'ontologies', network analysis for this project is both possible and promising.

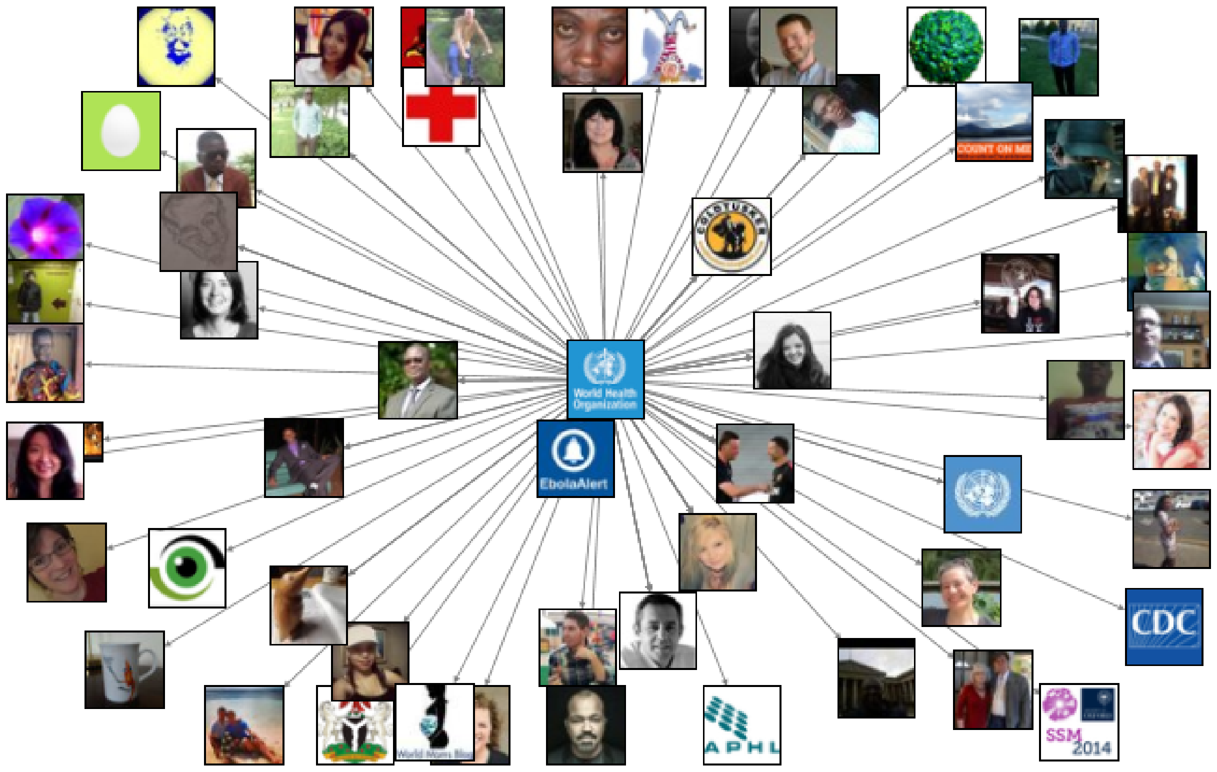


Figure 26: **Followers of the World Health Organization** (partial)

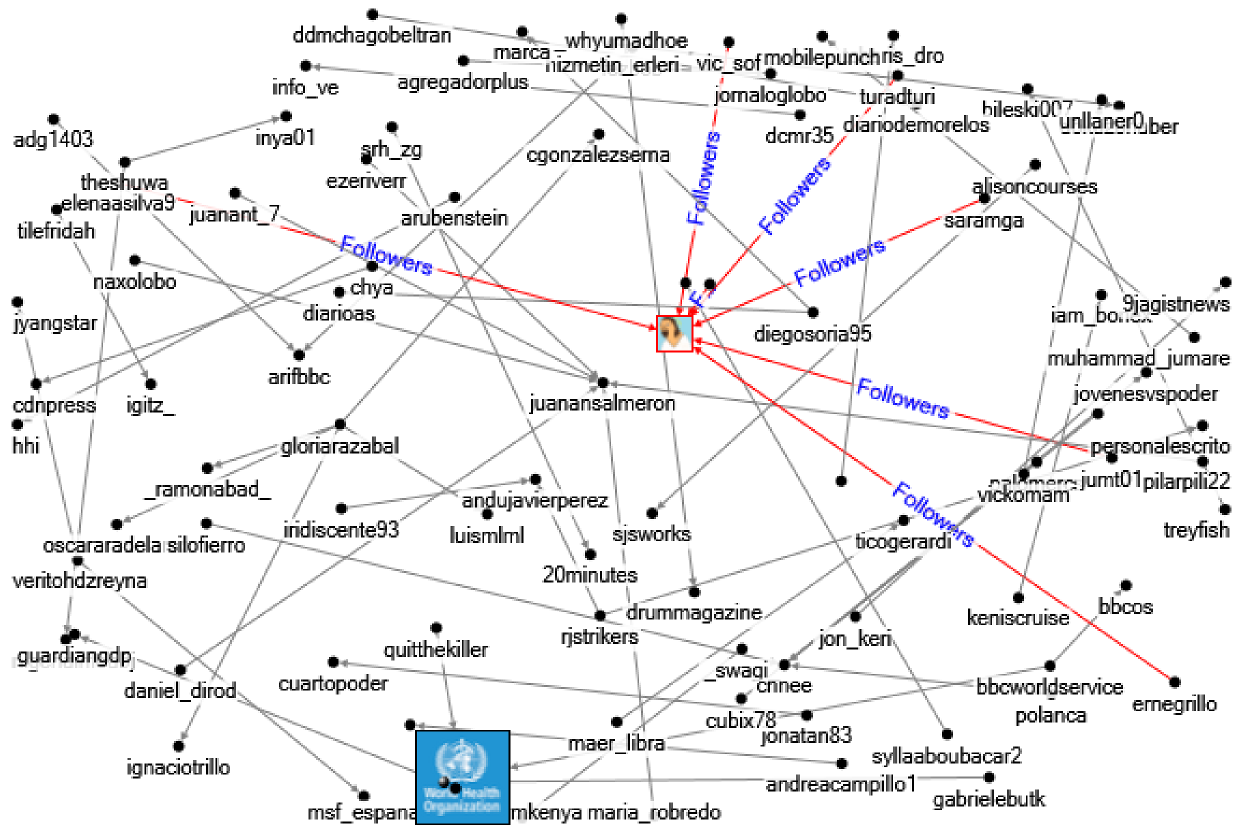


Figure 27: **People Tweeting About Ebola** (online, not in the dataset)

Part III
Other Research

13 Medicine-Related Twitter Projects

- How Twitter Is Studied in the Medical Professions:
  A Classification of Twitter Papers Indexed in PubMed
  [Williams et al., 2013a]

- What do people study when they study Twitter?
  [Williams et al., 2013b]

- Pandemics in the Age of Twitter:
  Content Analysis of Tweets during the 2009 H1N1 Outbreak
  [Chew and Eysenbach, 2010]

- The Use of Twitter to Track Levels of Disease Activity and Public Concern in the U.S. during the Influenza A H1N1 Pandemic
  [Signorini et al., 2011]

- The potential of social networks for early warning and outbreak detection systems:
  The swine flu Twitter study
  [Kostkova et al., 2010]

- Using Twitter and other social media platforms to provide situational awareness during an incident
  [Tobias, 2011]

- The other Twitter revolution:
  How social media are helping to monitor the NHS reforms
  [McKee et al., 2011]

- A visual backchannel for large-scale events
  [Dork et al., 2010]

- Dissemination of health information through social networks:
  Twitter and antibiotics
  [DScanfeld et al., 2010]

- Twitter as a communication tool for orthopedic surgery.
  [Franko, 2011]

- Machine intelligence for health information:
  Capturing concepts and trends in social media via query expansion
  [Su et al., 2011]

- Social Internet sites as a source of public health information
  [Vance et al., 2009]

- Hospitals are finding ways to use the social media revolution to raise money, engage patients and connect with their communities
  [Galloro, 2011]

- Twitter mining for fine-grained syndromic surveillance
  [syn, 2014]

- Now Trending #health In My Community
  [Department of Health and Human Services, 2012, US Dept. pf Health & Human Services, 2012]

- Physicians On Twitter
  [Sabine Tejpar et al., 2011]

- Agencies Use Social Media to Track Food-born Illnesses
  [BM, 2014]

- Social media in vascular surgery
  [Indes et al., 2013]

- Decoding Twitter:
  Surveillance and trends for cardiac arrest and resuscitation communication
  [Bosley et al., 2012]

- Twitter as a tool for ophthalmologists
  [Micieli and Micieli, 2012]

- Dissemination of health information through social networks
  Twitter and antibiotics
  [Scanfeld et al., 2010]

- All Atwitter About Radiation Oncology:
  A Content Analysis of Radiation Oncology-related Traffic on Twitter
  [Jhawar et al., 2012]

Appendix A Fields Added To Twitter json

This section shows examples of the json fields added to the full Twitter data by the python programs `get_twitter_json.py`, `update_geo_data.py` and `reverse_geocoding.py`, which can be found in the `code` folder on GitHub [Fisher, 2014b].

A link to download the consolidated file that contains all of these additional fields with all duplicates removed in gzip format can be found on page 40. See Appendix D on page 46 for examples of how to process this file using R, Python and MongoDB.

The official guide to Twitter's json structures is here: [Twitter, 2014b]. Anyone thinking of doing an analysis of Twitter data should be quite familiar with this.

Note: the "geo" field and the "coordinates" field show lat and lon in a different order. The seemingly-backwards order of "coordinates" is generally preferred because it's an X and Y ordering that is more comfortable for programming map grids.

```
Additional Fields in Twitter json
=================================

"timestamp": 1389010334.0              # unix timestamp for Twitter's ['
   created_at'] field

"user": {                              # provided by Twitter user
        "location": "New York City",
        },

"geo": {                               # derived from ["user"]["location"]
        "type": "Point",
        "coordinates": [40.730599,
        -73.986581]
    },

"geo_reverse": {                       # derived from ["geo"]; data from
   2010 US census
        "areacode": "212",
        "Land_Sq_Mi": 0.576,
        "county": "New York",
        "FIPS": "36061",
        "state_abbr": "NY",
        "country_code": "US",
        "Type": "",
        "city": "New York",
        "country": "United States",
        "zipcode": "10003",
        "state": "New York",
        "Pop_2010": 56024.0
    },

"topsy": {                             # fields from original dataset

    "firstpost_date": "01/06/14",
    "timestamp": 1388984400.0,         # unix timestamp for ["topsy"]["
   firstpost_date"] field

```

```
    "url": "http://twitter.com/primary_immune/status/420090415086198784",
    "score": 7.2846317,
    "trackback_author_nick": "primary_immune",
    "trackback_author_url": "http://twitter.com/primary_immune",
    "trackback_permalink": "http://twitter.com/Primary_Immune/status
   /420090415086198784",


    "file_counter": 2,                  # original dataset number and name
    "short_file_name": "Jan to May\\Blood\\Tweets_BloodCancer.csv"
}
```

Appendix B Details of the S3 Twitter json Data Files

Note: Anyone interested in beginning an analysis should use the **consolidated file with duplicates removed**. It can be downloaded in gzip format from the link on page 40. Examples of how to write code using this file can be found in Appendix D starting on page 46.

Original csv Files

The original dataset consisted of 896 csv files with 6,543,272 lines, located on Google Drive `https://drive.google.com/folderview?id=0B2io9_E3COquYWdlWjdzU3ozbzg&usp=sharing`

Twitter json Files

Three files were produced by the process to add the full Twitter json to the original data. All three are stored in gzip format on S3. In addition to the original json for each tweet, a new field was added `["topsy"]` containing information about the original file from which the record was drawn.

- `https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/bigtweet_file001.gz`
  1.48 Gb/12.7 Gb, 3,040,986 lines
- `https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/bigtweet_file329.gz`
  340 Mb/2.71 Gb, 651,317 lines
- `https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/bigtweet_file361.gz`
  793 Mb/6.55 Gb, 1,509,351 lines

The total number of json records is 5,201,654, which is 1,341,618 fewer than the original files, a loss of 20.50%, because either the original record had no id or Twitter rejected it.

Note: I did not search for duplicate records or spam.

geo-tagged files

After the files had all the data from the original tweet, the `["geo"]["coordinates"]` field was filled in with the latitude and longitude (in the latest Twitter API this field is deprecated so its use for this purpose adheres to the old API without conflicting with the new one).

- `https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_geotagged.gz`
  946 Mb/8.26 Gb
  2,023,501 lines in total
  2,023,438 last line with non-blank ["geo"] field
  1,626,053 lines (80.36%) with non-blank ["user"]["location"] entries
  1,156,825 lines (57.17%) with non-blank ["geo"] fields

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_geotagged2.gz
  489 Mb/4.05 Gb
  1,065,986 lines in total
  1,065,986 last line with non-blank ["geo"] field
  804,470 (75.47%) with non-blank ["user"]["location"] entries
  587,257 (55.09%) with non-blank ["geo"] fields

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_geotagged3.gz
  496 Mb/4.11 Gb
  1,017,500 lines in total
  1,017,500 last line with non-blank ["geo"] field
  803,175 (78.94%) with non-blank ["user"]["location"] entries
  658,041 (64.67%) with non-blank ["geo"] fields

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_geotagged4.gz
  559 Mb/4.73 Gb

reverse geo-tagged files

These files were created using the latitude and longitude to add the field `["geo_reverse"]`, which is wholly outside of the Twitter API specification.

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_reversegeo.gz
  992 Mb/8.70 Gb

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_reversegeo2.gz
  512 Mb/4.28 Gb

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_reversegeo3.gz
  522 Mb/4.33 Gb

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_reversegeo4.gz
  587 Mb/4.98 Gb

consolidated file with duplicates removed

This file is the only file an analyst should use because it contains all the Twitter data plus all the additional data shown in Appendix A on page 37 plus some minor improvements to the geo-tagging and reverse geo-tagging algorithms minus all duplicate records. See Appendix D on page 46 for examples of using this file in R and Python.

- https://s3-us-west-1.amazonaws.com/data.healthcare-twitter-analysis.com/HTA_noduplicates.gz
  1.85 Gb zipped / 15.80 Gb unzipped

The original Topsy csv files contained 6,543,272 lines, 1,341,618 of which were rejected as invalid tweets by Twitter; of those remaining, another 1,465,937 were found to be duplicates (same Twitter id), leaving 3,783,557 tweets in this final dataset.

3,783,557 total distinct tweets
1,341,928 tweets with US country codes
911,716 tweets with non-US country codes
238 total country codes

Process Summary

1. To derive the Twitter data, the original data was processed in batches of 100 tweets, a limit imposed by Twitter for automated requests, and a text file was appended and saved after each batch was processed.

   The process of running the program is fairly fast: on a Dell Windows 7 laptop it processed about 1,700 tweets per minute; however, the elapsed time was much longer because Twitter imposes a limit of around 14,000 tweets per 15-minute interval, so the program goes to sleep every 13,500. On Amazon's EC2, the program to add Twitter's json ran to completion in 8 days, 2 hours.

2. The process to add the ["geo"] coordinates is similar: if the field ["geo"] exists, it is accepted unchanged; otherwise the ["user"]["location"] field is parsed and MapQuest queried for lat/lon which are put into the ["geo"]["coordinates"] field.

3. Finally, the ["geo"]["coordinates"] field of every record was queried and an algorithm used to locate the record in a country; if the country was the United States, into a city, zipcode, county and state. Information about the zipcode was added from the 2010 US census, as well as the county FIPS code which is required for mapping at the county level.

All the programs involved in this process plus their supporting files and databases can be found on the GitHub repo [Fisher, 2014b].

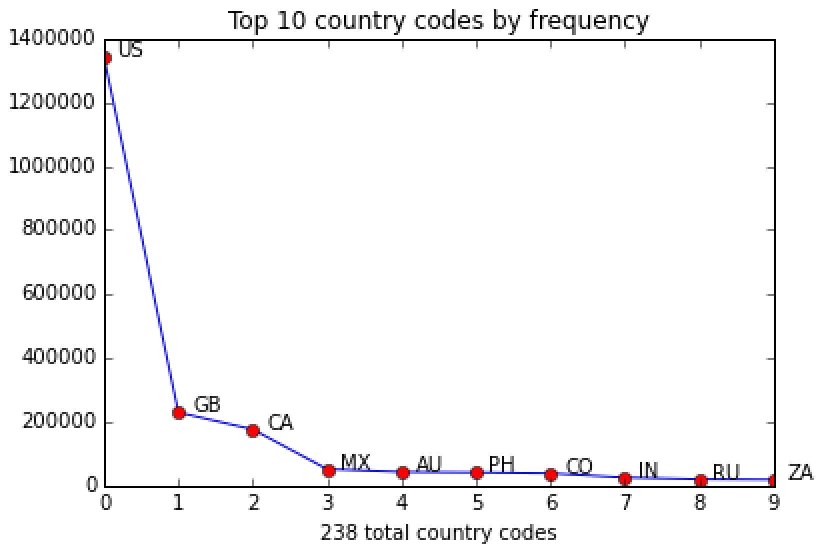


Figure 28: **Top 10 Country Codes**

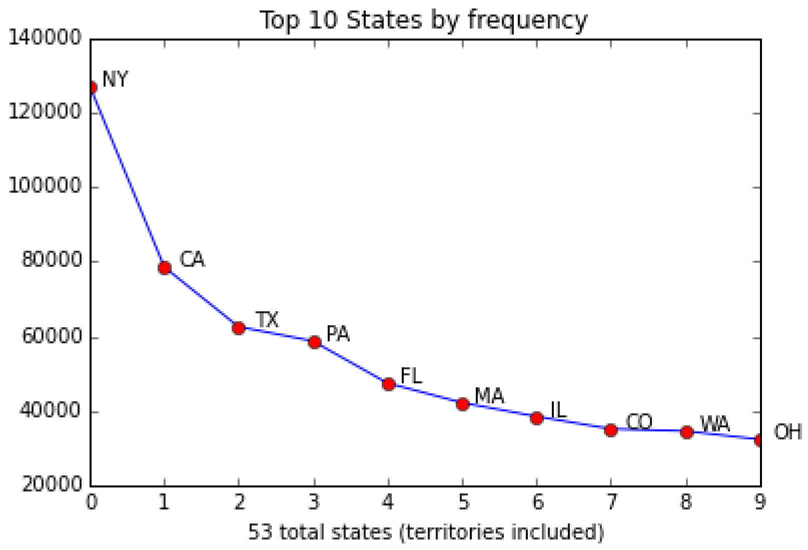


Figure 29: **Top 10 States**

Appendix C Amazon Web Services EC2 & S3

AWS EC2 and S3 have rather obscure documentation and operate in basic command-line mode, however once you've mastered them they are quite useful since you can get essentially as much computing power, storage and Internet access as you could possibly need on demand.

- EC2 is the name for the service that provides either Unix or Windows servers on demand.
- S3 is the name of bit-bucket data storage.

On top of the base operating system you have to build your own programming environment. I used IPython, see pages 44 and 45.

In addition to being quite useful, it is also inexpensive: even with numerous false starts and long run times my total bill for this project was less than $50.00.

Amazon Web Services for background Python

I assume you have an AWS account and an access Key pair for SSH access. On Windows I used Putty as my SSH terminal and WinSCP for FTP; on my iPhone I used Server Auditor.

Setup:

1. I started an EC2 Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-e7b8c0d7 on an x86_64 t2.micro configuration. The SSH logon for such an image is ubuntu@Public IP.

2. I struggled for an entire day trying to figure out how to access files on S3 from EC2; I gave up and FTP'd the entire bunch from Google Drive to the image I had just started. There is a nice tool at http://timkay.com/aws/ that is helpful but not for 897 files in recursive folder structures.

3. I also had to download

    1. `get_twitter_json.py` and edit it a little for Ubuntu file formats
    2. `mapquest_key.txt`
    3. `twitter_credentials.py`
    4. `twitter_functions.py`
    5. `filename_list.csv` had to be re-created for the Ubuntu file names and locations

Install Python:

1. I downloaded the Anaconda distro:
   `wget http://09c8d0b2229f813c1b93-c95ac804525aac4b6dba79b00b39d1d3.r79.cf1.rackcdn.com/Anaconda-2.0.1-Linux-x86_64.sh`

2. ... and installed it
   `bash Anaconda-2.0.1-Linux-x86_64.sh`
   Note: 'q' gets you out of the license agreement

3. Reloaded the .bashrc ...
   `source .bashrc`

4. ... and issued the following commands:
   `sudo -i`
   `apt-get update`
   `apt-get install python-pip`
   `pip install oauth2`
   `apt-get install ipython`

Then I started up the python program in the background

```
nohup python get_twitter_json.py "filename_list.csv" 1 0 &
```

... and exited the shell
`exit`

As the program churned through the files I was able to sign on and monitor progress via the `nohup.out` file. I could also watch system statistics through the AWS Management Console and on the iPhone AWS app. I probably could have used `boto` but I didn't try it.

Create S3 zip file

The first step is to compress it:

```
infilename  = 'HTA_geotagged.json'
outfilename = 'HTA_geotagged.gz'

import gzip
f_in  = open(infilename, 'rb')
f_out = gzip.open(outfilename, 'wb')
f_out.writelines(f_in)
f_out.close()
f_in.close()
```

... and then to move it to S3

Install utilities from `http://timkay.com/aws/`

```
 * sudo -i
 * apt-get install curl
 * curl https://raw.githubusercontent.com/timkay/aws/master/aws -o aws
 * vi ~/.awssecret # AWS credentials Ctrl+o :w <enter> Ctrl-o :q <enter>
 * perl aws --install
 * chmod +x aws
 * cd /home/ubuntu
```

Then you can enter `s3put <S3 bucket name> <local file to be transferred into S3>`

SQLITE3 is required for my reverse geo-tagging functions: `sudo apt-get install sqlite3 libsqlite3-dev`

Appendix D Sample Programs

The individual tweets in the `HTA_noduplicates.json` file can be accessed as follows:

D.1 R

```r
#
# Note: in all my analyses I used Python to read the json file
#       and created csv files with the specific data I needed for
#       the analyses I wanted to do in R.
#
#       Loading 4 million records into an R data.frame
#       is not a good idea: either it won't work at all because
#       of the configuration of your machine or else it will be
#       horribly slow.
#
#       Nontheless, on a reasonable subset, it can be done
#       and many packages are available to work with the
#       Twitter json record.
#
library(rjson)
file_path  = ("HTA_noduplicates.json")
tweet_list = fromJSON(sprintf("[%s]", paste(readLines(file_path),collapse=
    ",")))

retweets  = tweet_list[[i]]$retweet_count
user_name = tweet_list[[i]]$user$name
text      = tweet_list[[i]]$text

for (i in 1:length(tweet_list)){
    if (retweets > 100){
        cat(sprintf("\n\n%d %s\n%s",retweets, user_name, text))
    }
}
## convert to twitteR structure
library(twitteR)
tweets = import_statuses(raw_data=tweet_list)
```

Listing 1: Read a json file with R

D.2 Python

Python was the workhorse of this report: the datasets were all created with Python and the analyses were either done directly in Python or else Python was used to create CSV-file inputs to R.

```python
import json, datetime

with open("HTA_noduplicates.json", "r") as tweet_file:
    # go line-by-line converting each into a json tweet
    for line in tweet_file:
        tweet     = json.loads(line)

        # look for mentions of depression
        text      = tweet['text']
        if 'depres' not in text.lower(): continue

        # pull out the fields of interest
        user_name = tweet['user']['name']

        ISO2_cc   = tweet['geo_reverse']['country_code']
        city      = tweet['geo_reverse']['city']
        county    = tweet['geo_reverse']['county']
        FIPS      = tweet['geo_reverse']['FIPS']
        state     = tweet['geo_reverse']['state']
        zipcode   = tweet['geo_reverse']['zipcode']

        orig_file = tweet['topsy']['short_file_name']

        timestamp = tweet['timestamp']
        date_str  = datetime.datetime.fromtimestamp(int(timestamp)). \
            strftime('%a %b %d, %Y %H:%M:%S')

        # print tweets that have a country code
        if ISO2_cc != "":
            print "\n(%s) %s %s %s %s %s %s"% \
                (user_name, city, county, FIPS, state, zipcode, ISO2_cc)
            print "%s"%(text)
            print "%s %s"%(date_str, orig_file)
```

Listing 2: Read json file with Python

```
(My Foggy Brain) Rancagua     CL
And the Acting Award Goes To.... YOU! Yes, You with the Chronic Pain!!
http://t.co/8QdlbExg5u #Fibromyalgia #Depression #Tremor
Tue May 06, 2014 03:04:45 /Jan to May/Musculoskeletal/Tweets_Fibromyalgia.
    csv

(Federal Practitioner) New York New York 36061 New York 10003 US
#NSAIDs may reduce #depression in patients with #osteoarthritis.
    @NSAIDAware http://t.co/sY4fjbAtrt
Tue Apr 08, 2014 14:40:46 /Jan to May/Musculoskeletal/Tweets_
    Osteoarthritis.csv

```

```
(Debbie Cook) Tinglev     DK
#NASS #Helpline been v busy today - discussions on #AntiTNF, #depression,
    foot problems, fractures &amp; #pregnancy #ankylosing #spondylitis
Mon Feb 03, 2014 09:35:19 /Jan to May/Musculoskeletal/Tweets_Spondylitis.
    csv

(Laurent BOGHOSSIAN) Laguna Niguel Orange 06059 California 92607 US
RT @FemaleDepressed: #add #psychology #oola #depression #mentalhealth
Mon Jan 13, 2014 03:16:26 /Jan to May/Neurological/Tweets_ADD.csv

(Jennifer Thompson) Kirribilli     AU
After cancer - depression... #breastcancer http://t.co/BmrZFzjTOX
Thu Feb 06, 2014 13:54:33 /Jan to May/Other Files/Tweets_Breastcancer.csv

(Online Therapist) Louisville Jefferson 21111 Kentucky 40217 US
Talk Therapy Online - Effective &amp; Convenient help for #ANXIETY &amp; #
    DEPRESSION. http://t.co/HlsKl0jBmY. Email me. Please Retweet!
Tue Feb 25, 2014 15:13:36 /Jan to May/Other Files/Tweets_Depression.csv

(Ula Hynds) Effingham Florence 45041 South Carolina 29541 US
#Depression &amp; #Anxiety can cause sleep disturbances.  Take a free
    screening at http://t.co/jw1Pht0gT7
Tue Feb 25, 2014 15:09:44 /Jan to May/Other Files/Tweets_Depression.csv
```

Listing 3: Sample result of Python program

Notice the files from which these mentions of depression were drawn; the point is that you probably do not want to restrict yourself to someone else's categorization.

D.3 MongoDB

D.3.1 Load The File on an External Hard Drive

On my Windows 7 laptop the `E:` drive is a one-terabyte external drive, in the `HTA` folder of which is the 15.5 Gb `HTA_noduplicates.json` file that I g-unzipped from a download from S3 on AWS. To bulk insert this file into a MongoDB database collection on the same external hard drive I followed this procedure:

I started one terminal using the "Run as Administrator" right-click option on the `cmd.exe` terminal program and entered two commands (1) to stop the default MongoDB Windows service which uses databases on my C drive and (2) start the server using the external drive:

```
> net stop MongoDB
> mongod --dbpath "E:\HTA"
```

On a second terminal instance I entered

```
> mongoimport --db HTA --collection grf --type json --file E:\HTA\HTA_
   noduplicates.json
```

This ran for 5 minutes or so and created files named HTA.0 ... HTA.14, HTS.ns
I then entered

```
> mongo
> show dbs

HTA     21.943GB
admin    (empty)
local    0.078GB

> use HTA
> show collections

grf
system.indexes

> db.grf.stats()
{
        "ns" : "HTA.grf",
        "count" : 3783557,
        "size" : 20826900400,
        "avgObjSize" : 5504,
        "storageSize" : 21564321680,
        "numExtents" : 32,
        "nindexes" : 1,
        "lastExtentSize" : 2146426864,
        "paddingFactor" : 1,
        "systemFlags" : 1,
        "userFlags" : 1,
        "totalIndexSize" : 122787168,
        "indexSizes" : {
                "_id_" : 122787168
        },
        "ok" : 1
}
```

... 3,783,557 documents in the collection which is 21,564,321,680 bytes in size, each document in which is an average of 5,504 bytes; at this point there is only one index, the default on the "_id" inserted automatically for us during the insert process (it might be more efficient to modify the json text records to change the Twitter id field from "id" to "_id" to take advantage of this index ... since they are all by this time unique this would work).

... and finally,

```
> db.grf.findOne()
```

produced the json for one tweet as follows (abbreviated):

```
{
        "_id" : ObjectId("542d8979699cad0f8ce1b360"),
        "contributors" : null,
        "truncated" : false,
        "text" : "These groups have joined forces in the #
    bleedingdisorders community to advance #hemophilia research: http://t.
    co/5blIDLpeaH",
        "in_reply_to_status_id" : null,
        "id" : NumberLong("419103391600488450"),
        "favorite_count" : 1,
        "topsy" : {
                "file_counter" : 1,
                "firstpost_date" : "01/03/14",
                "url" : "http://twitter.com/nhf_hemophilia/status/
    419103391600488450",
                "timestamp" : 1388707200,
                "score" : 11.115716,
                "trackback_author_nick" : "nhf_hemophilia",
                "trackback_author_url" : "http://twitter.com/nhf_
    hemophilia",
                "trackback_permalink" : "http://twitter.com/NHF_Hemophilia
    /status/419103391600488450",
                "short_file_name" : "/Jan to May/Blood/Tweets_
    BleedingDisorders.csv"
        },

(...)

        "geo" : {
                "type" : "Point",
                "coordinates" : [
                        40.730599,
                        -73.986581
                ]
        },

        "geo_reverse" : {
                "city" : "New York",
                "areacode" : "212",
                "country" : "United States",
                "Land_Sq_Mi" : 0.576,
                "zipcode" : "10003",
                "county" : "New York",
                "state" : "New York",
                "FIPS" : "36061",
                "state_abbr" : "NY",
                "country_code" : "US",
                "Pop_2010" : 56024,
                "Type" : ""
        },
        "place" : null
}
```

D.3.2 Process the MongoDB Database with pymongo

```
import json
from pymongo import MongoClient

# start up MongoDB
# ================
client = MongoClient()  # after mongod --dbfile "E:\HTA"

db     = client['HTA']  # reference the database
tweets = db.grf         # reference the collection

for tweet in tweets.find():
    if tweet['geo'] and tweet['coordinates']:
        print tweet['text']
        print 'location\n',tweet['user']['location']
        print 'geo\n',json.dumps(tweet['geo'],indent=4)
        print 'coordinates\n',json.dumps(tweet['coordinates'],indent=4)
        print 'geo_reverse\n',json.dumps(tweet['geo_reverse'],indent=4)
```

Listing 4: MongoDB from Python:
read and process the data

```
Still in DC reflecting on the great #nhfwd14 #hemophilia &amp; #
   bleedingdisorders advocacy. #mainehemophilia
location
Portland, Maine
geo
{
    "type": "Point",
    "coordinates": [
        38.8581341,
        -77.05241167
    ]
}
coordinates
{
    "type": "Point",
    "coordinates": [
        -77.05241167,
        38.8581341
    ]
}
geo_reverse
{
    "city": "Arlington",
    "areacode": "703",
    "country": "United States",
    "Land_Sq_Mi": "",
    "zipcode": "22240",
    "county": "Arlington",
    "state": "Virginia",
    "FIPS": "51013",
    "state_abbr": "VA",
    "country_code": "US",
    "Pop_2010": "",
    "Type": "U"
}
```

Listing 5: Sample result of Python program using MongoDB

This is a person whose location says they're from Maine, a member of the Maine Hemophilia Association, but who is tweeting from the DC area while attending the National Hemophilia Fund 2014 conference.

The fact the this tweet contains a "coordinates" field means that the latitude and longitude were created by their phone's GPS and therefore are quite reliable.

D.3.3 Indexes

This example does not take advantage of MongoDB's indexing facility which would certainly improve search performance over the simple Python search I have illustrated. Only 16 indexes are permitted per collection so the choice of what to index will be application dependent.

Indexes are the single biggest determinant of search & sort efficiency but if possible you want them to all fit in memory so their size is important ... you can't just willy-nilly add indexes and expect performance to improve if you start paging. The size of the single default index we have is shown as follows:

```
> db.grf.totalIndexSize()
122787168
```

122,787,168 ... 123 Megs per index (more for multi-key indexes, potentially less for a sparse index) can add up.

'retweet_count' is commonly used to assign a value or a priority to a tweet: the higher the retweet count, the greater the value. Please note: 'retweeted' is always false; look for true and you will scan every record in the database and fail.

```
> db.grf.ensureIndex( { "retweet_count" : 1 } )
```

Searching for specific country codes is also very common and an index might help performance:

```
> db.grf.ensureIndex( {"geo_reverse.country_code": 1} )
```

... compound searches for both country and state can be optimized as follows, in which case, since the country_code is first, it will also optimize just country_code searches as well:

```
> db.grf.ensureIndex( {"geo_reverse.country_code": 1, "geo_reverse.state_
   abbr": 1})
```

D.3.4 Geo Spatial Indexes

Note that the ”geo” field and the ”coordinates” field in the example above show lat and lon different orders. The seemingly-backwards order of ”coordinates” is preferred because it’s an X and Y ordering that is more comfortable for programming map grids. The reason Twitter has deprecated the ”geo” field is because it was the wrong way for the rest of the geo-locating world, including MongoDB.

To keep from departing too far from Twitter’s documentation I have maintained the order in the ”geo” field which is where I stored the imputed coordinates. This means that geo spatial indexing as shown below will not work with the ”geo” field without modification, but go ahead and feel free to modify if it will help you since there were very few ”coordinates” fields in the original dataset but nearly 60% of the records have ”geo” fields as a result of my coordinate-imputation algorithm.

To create a geo-spatial index in MongoDB using the mongo command line you would say

```
> db.grf.ensureIndex({coordinates: "2d"})
```

Then to find the 10 nearest locations to a given X (lon) and Y (lat) location in increasing order of distance you would say

```
> db.grf.find({coordinates: {$near : [X,Y]}}).limit(10)
```

You can get very elaborate as you experiment with the data, for example creating 2d indexes that also include hashtags and so on.

Look up the Geo JSON standard and `2dsphere` indexing in MongoDB if you get excited by all the possibilities offered by this.

D.3.5 Full Text Search Indexing

In some of my examples for text analysis I parsed the `text` field into tokens (hashtags, user-mentions, URLs and words) using a helper function I wrote in Python and operated on individual tokens. An alternative is to establish a MongoDB `text` index.

```
> // the field to be indexed is on the left
> // the field can be any field, 'user.location' might be another choice
> db.grf.ensureIndex({'text' : 'text'})
```

Having set up the index, you can search as follows

```
> db.grf.find({$text: {$search: '#sepsis'}})
```

There’s lots more, such as searching on several terms, an implicit OR operation, and ranking the ’quality’ of the documents returned. The best match would contain all the terms, lower scores would have fewer than all the terms.

D.3.6 RESTful Interface

pymongo gives a programmatic interface into a MongoDB database from Python but it's also useful to have an HTTP REST interface so you can serve data to a web page. The fact that JavaScript on the web page and MongoDB both use json is very helpful in this regard. Python has several HTTP modules, the simplest of which is Bottle.

I may include more details in the repo if I develop the interface further but here is a basic skeleton (drawn from actual working code on my computer) ... the Bottle route for making generalized MongoDB queries looks like this

```
import pymongo, bottle
from bottle import error, route, get, post, static_file, request, abort,
    response
import cgi, re, os, json
import bson.json_util

                ...

@route('/query/<limit>', method="POST")
def query(limit):
    data = request.json;

    # execute the MongoDB query, returning a list of the _id's that match
    if int(limit) <= 0:
        id_list = [tweet for tweet in tweets.find(data,dict({"_id":1}))]
    else:
        id_list = [tweet for tweet in tweets.find(data,dict({"_id":1})).
    limit(int(limit))]

    # how long is the list?
    num = len(id_list)

    # also include the first result
    example = tweets.find_one(data)

    # create the structure to return
    result                = dict()
    result['num']         = num
    result['id_list']     = id_list
    result['example']     = example

    # convert from MongoDB json to strict json and return the result
    result = bson.json_util.dumps(result)
    return result

    ...

connection_string = "mongodb://localhost"
connection        = pymongo.MongoClient(connection_string)

db     = connection.HTA
tweets = db.grf

```

```
# for production use: remove bottle.debug entirely and remove , reloader=
    True from bottle.run
bottle.debug(True)
bottle.run(host='localhost', port=8082, reloader=True)
```

Listing 6: Bottle route for a RESTful interface

This Bottle code will take in a json MongoDB query exactly as it would be entered on the command line plus a limit (where 0 means retrieve all results); the json is used to query the database and a structure is returned consisting primarily of a list of the '_id's of the tweets that matched (rather than sending all the data at once, the idea is to buffer requests as the user pages through the results.)

The jQuery code in the web page to call the Bottle REST interface is as follows ...

```
        // create the query as a stringified json object plus the limit
        // ------------------------------------------------------------
        var query_obj = {$text: {$search: '#Sepsis'}};
        var query_str = JSON.stringify(query_obj);
        var limit = 5;

        // send the query to the server
        // ----------------------------
        var request = $.ajax({
          url:          "http://localhost:8082/query/" + limit,
          type:         "POST",
          data:         query_str,
          dataType:     "json",
          contentType: "application/json"
        });
        request.done(function( response ) {
            /* if the request is successful ...
               do something with the response
               which will be the json "structure"
               { "num":      the number of _id's,
                 "id_list": [_id, _id, ...],
                 "example": { tweet } }            */
        });
        request.fail(function( jqXHR, textStatus ) {
          // if the request fails ...
        });
```

Listing 7: jQuery ajax call to the Bottle route

D.3.7 Databases Over Multiple Disks

There is a post in Stack Overflow on how to assign MongoDB databases to different disk drives simultaneously without stopping the process as I showed in the example above: [StackOverflow, 2014].

Appendix E Notes on Geo Data

Twitter provides the ability for a user to opt in to tagging their tweets with GPS data; the `tweet["geo"]`, `tweet["place"]` and `tweet["coordinates"]` fields contain this data when provided. However, in our dataset fewer than 3% of the records have anything but `null` in these fields.

E.1 Geo Tagging

Without GPS data we are left with only the `tweet["user"]["location"]` field which is entered as text by users when they first set up their account and when it is not blank, it often contains either outright junk or, at best, apparently-correct information in a highly-unstructured format that has to be laboriously parsed to provide anything useful to a program attempting to produce a geographic-based analyses.

One example of the problem is the fact that MapQuest [Mapquest, 2014] returns 47 choices when presented with 'Pasadena'. As a fan of Jan & Dean, my initial thought was that the city in California near Los Angeles was probably intended most of the time, but it turns out that Pasadena, Texas has a larger population. 'Swansea' might make you think of Wales, but there are Swanseas in Australia, Canada and the United States; founded by Welsh emigres, no doubt.

`tweet["user"]["utc_offset"]` and/or `tweet["user"]["time_zone"]` could be used in an improved version to narrow in on the longitude of the tweet from among a number of choices, but that was not done here.

Another problem type is 'Cleveland, OH - San Diego, CA' or 'Vancouver & Caracas', both real examples. The locations are legitimate but indecipherable when looking to connect a tweet to a single point on the globe.

The program `update_geo_data.py` in the repo is my most-recent attempt to crack this code. You are welcome to use it, learn from it or critique it (use the Issues tab of GitHub).

E.2 Reverse Geo Tagging

The step following the effort to assign latitude and longitude coordinates is called *reverse geo-tagging* and involves using the coordinates to derive the city and country; and when the country is the United States ... zipcode, city, county, FIPS and state. It is unlikely that I will attempt to collect the detailed information for countries other than the United States, although the data is available and my algorithms are easily extensible.

E.3 Current State of Twitter Geo Tagging

I have spot checked a fairly large subset of our dataset and my conclusion is that the results are good enough for exploratory analyses, to test algorithms and to generate hypotheses; surprisingly good given the raw material, but nowhere near good enough to make actual medical-related decisions as the data stands right now.

My location-parsing process can be improved upon but until it is, and really until a much larger number of users opt in for GPS tagging, this data must be considered experimental.

What is lacking is the basic lat/lon of the user; the reverse geo-coding process is pretty straightforward and much less subject to criticism –give me a lat/lon and I can very accurately tell you the zipcode– but the fundamental two bits of information, the lat and lon, are simply not reliable at the moment.

A number of the other Twitter studies I have cited claim to be able to track tweets in the spatial as well as temporal dimension; I am very doubtful they can do so accurately.

Appendix F Notes on Big Data and Parallel Processing

To the worlds of astronomy, genomics or search engines our dataset isn't very big; but at 6 million records it's big enough to require special handling ... it took quite a long time to process it as it was and had it been run serially on a single machine (which was my initial approach), it would still be running.

Hadoop, Pig and so forth are all the rage these days but parallel processing is possible on as small and simple a configuration as a single core machine running a modern multi-tasking operating system; a multi-core machine will, of course, perform even better.

Assigning each task to run exclusively on its own dedicated core or machine is the default idea of parallel processing but any multi-tasking operating system will do a pretty good job of using interrupts to assign tasks to available resources without requiring the programmer to do anything more complex than split the data and programs into discrete parts.

If several tasks are run simultaneously in the background[2] their processing will be interleaved by the operating system. The three main steps of processing our dataset (1) getting Twitter json (2) geo-tagging and (3) reverse geo-tagging, as currently designed, are inherently serial for each record. But within each step the data can be processed in any order; in addition, if a part of one step finishes before the other parts of the same step, the subsequent step can begin on the result.

Clearly, there is some threshold of tasks-to-cores after which the process as a whole is slower than serial but it is a function of which resource is constrained; two CPU-bound processes may not be able to play nicely on a single core but I/O bound jobs probably could. One of the advantages of a service like AWS is that you can spin up whatever configuration you need: lots of disk space, lot of RAM, many cores; or not, depending on your needs ... you can scale **up** and/or you can scale **out**. I did both in this project.

If later somebody comes up with new data to add to each record, the same poor-man's map/reduce technique can be applied.

[2]On a Linux machine, to run a task in the background you preface the startup command with 'nohup' and suffix it with '> console_file.out &'. This will start the process in the background and you will be able to sign on later and look at the console output by saying something like 'tail console_file.out' as long as the *.out files have unique names for each background process.

References

[syn, 2014] (2014). Twitter mining for fine-grained syndromic surveillance. Artifical Intelligence in Medicine 61 (2014) 153-163.

[BioPortal, 2014] BioPortal (2014). Bioportal, comprehensive repository of biomedical ontologies. http://bioportal.bioontology.org/.

[BM, 2014] BM, K. (2014). Agencies use social media to track foodborne illness. *JAMA*, 312(2):117–118.

[Bosley et al., 2012] Bosley, J. C., Zhao, N. W., Hill, S., Shofer, F. S., Asch, D. A., Becker, L. B., and Merchant, R. M. (2012). Decoding twitter: Surveillance and trends for cardiac arrest and resuscitation communication. http://www.resuscitationjournal.com/article/S0300-9572(12)00871-4/abstract.

[Breen, 2011a] Breen, J. (2011a). Github twitter-sentiment-analysis-tutorial-201107. https://github.com/jeffreybreen/twitter-sentiment-analysis-tutorial-201107. Code provided in conjunction with tutorial slides.

[Breen, 2011b] Breen, J. (2011b). slides from my r tutorial on twitter text mining #rstats. http://jeffreybreen.wordpress.com/2011/07/04/twitter-text-mining-r-slides/.

[Chew and Eysenbach, 2010] Chew, C. and Eysenbach, G. (2010). Pandemics in the age of twitter: Content analysis of tweets during the 2009 h1n1 outbreak. http://www.plosone.org/article/info%3Adoi%2F10.1371%2Fjournal.pone.0014118.

[CodePlex, 2014] CodePlex (2014). Nodexl: Network overview, discovery and exploration for excel. https://nodexl.codeplex.com/. An Excel template that provides for network analysis and visualization.

[Computerworld, 2010] Computerworld (2010). Twitter growth prompts switch from mysql to 'nosql' database. http://www.computerworld.com/s/article/9161078/Twitter_growth_prompts_switch_from_MySQL_to_NoSQL_database.

[Cook, 2014] Cook, T. (2014). Tim Cook healthcare twitter analysis repo. https://github.com/twcook/TweetMapping.

[Department of Health and Human Services, 2012] Department of Health and Human Services (2012). Now Trending: #Health in My Community. http://nowtrending.hhs.gov/. This contest challenged entrants to create a web-based application that searched open source Twitter data for health topics and delivered analyses of that data for both a specified geographic area and the national level.

[Dork et al., 2010] Dork, M., Gruen, D., Williamson, C., and Carpendale, S. (2010). A visual backchannel for large-scale events. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5613451.

[DScanfeld et al., 2010] DScanfeld, Scanfeld, V., and Larson, E. (2010). Dissemination of health information through social networks: Twitter and antibiotics. http://www.ajicjournal.org/article/S0196-6553(10)00034-9/abstract.

[Fisher, 2014a] Fisher, G. (2014a). Files with twitter json added. https://s3-us-west-2.amazonaws.com/healthcare-twitter-analysis/bigtweet_file001.gz https://s3-us-west-2.amazonaws.com/healthcare-twitter-analysis/bigtweet_file329.gz https://s3-us-west-2.amazonaws.com/healthcare-twitter-analysis/bigtweet_file361.gz. These files contain the full Twitter json for the records provided by Topsy.

[Fisher, 2014b] Fisher, G. (2014b). George Fisher healthcare twitter analysis github repo. https://github.com/grfiv/healthcare_twitter_analysis.

[Franko, 2011] Franko, O. (2011). Twitter as a communication tool for orthopedic surgery. http://www.ncbi.nlm.nih.gov/pubmed/22050252?dopt=Abstract.

[Galloro, 2011] Galloro, V. (2011). Hospitals are finding ways to use the social media revolution to raise money, engage patients and connect with their communities. http://www.ncbi.nlm.nih.gov/pubmed/21513035?dopt=Abstract.

[Google Drive, 2014] Google Drive (2014). Healthcare twitter analysis data files. https://drive.google.com/folderview?id=0B2io9_E3COquYWdlWjdzU3ozbzg&usp=sharing.

[highscalability.com, 2011] highscalability.com (2011). How twitter stores 250 million tweets a day using mysql. http://highscalability.com/blog/2011/12/19/how-twitter-stores-250-million-tweets-a-day-using-mysql.html.

[Indes et al., 2013] Indes, J. E., Gates, L., Mitchell, E. L., and Muhs, B. E. (2013). Social media in vascular surgery. http://www.jvascsurg.org/article/S0741-5214(12)02104-0/abstract.

[Jhawar et al., 2012] Jhawar, S., Sethi, R., Yuhas, C., and Schiff, P. (2012). All atwitter about radiation oncology: A content analysis of radiation oncology-related traffic on twitter. http://www.redjournal.org/article/S0360-3016(12)02712-5/abstract.

[Kostkova et al., 2010] Kostkova, P., de Quincey, E., and Jawaheer, G. (2010). The potential of social networks for early warning nad outbreak detection systems: the swine flu twitter study. http://ijidonline.com/article/S1201-9712(10)00507-2/abstract.

[Mapquest, 2014] Mapquest (2014). Mapquest developer api. http://developer.mapquest.com/.

[McKee et al., 2011] McKee, M., Cole, K., Hurst, L., Aldridge, R., and Horton, R. (2011). The other twitter revolution: how social media are helping to monitor the nhs reforms. http://www.ncbi.nlm.nih.gov/pubmed/21325389?dopt=Abstract.

[Mehta and Saama Technologies, 2013] Mehta, P. and Saama Technologies (2013). Healthcare twitter analysis website. https://www.coursolve.org/need/184.

[Micieli and Micieli, 2012] Micieli, R. and Micieli, J. A. (2012). Twitter as a tool for ophthalmologists. http://www.canadianjournalofophthalmology.ca/article/S0008-4182(12)00294-3/abstract.

[MongoDB, 2014] MongoDB (2014). Mongodb website. http://www.mongodb.org/.

[Neo4j.org, 2014] Neo4j.org (2014). Linked data: Rdf and sparql with neo4j. http://www.neo4j.org/develop/linked_data. A graph database that holds RDF-like triples and supports SPARQL queries.

[Nielsen, 2011] Nielsen, F. Å. (2011). Afinn. http://www2.imm.dtu.dk/pubdb/views/bibtex.php?id=6010. Informatics and Mathematical Modelling, Technical University of Denmark.

[Quora, 2012] Quora (2012). Twitter: Which database system(s) does twitter use? https://www.quora.com/Twitter-1/Which-database-system-s-does-Twitter-use.

[Sabine Tejpar et al., 2011] Sabine Tejpar, MD, P., Wendy De Roock, MD, P., and Derek Jonker, M. (2011). Physicians on twitter. *JAMA*, 305(6):566–568.

[Sanchez, 2012] Sanchez, G. (2012). Mining twitter. https://github.com/gastonstat/Mining_Twitter. An interesting collection of R programs to do analyses of Twitter data. His use of ggplot was out of date but aside from fixing that, the R code worked pretty well in the context of the data for this project.

[Scanfeld et al., 2010] Scanfeld, D., Scanfeld, V., and Larson, E. L. (2010). Dissemination of health information through social networks: Twitter and antibiotics. http://www.ajicjournal.org/article/S0196-6553(10)00034-9/abstract.

[Signorini et al., 2011] Signorini, A., Segre, A. M., and Polgreen, P. M. (2011). The use of twitter to track levels of disease activity and public concern in the u.s. during the influenza a h1n1 pandemic. http://www.plosone.org/article/info%3Adoi%2F10.1371%2Fjournal.pone.0019467.

[StackOverflow, 2014] StackOverflow (2014). How to configure mongo to use different volumes for databases? https://stackoverflow.com/questions/24811046/how-to-configure-mongo-to-use-different-volumes-for-databases.

[Su et al., 2011] Su, X., Suominen, H., and Hanlen, L. (2011). Machine intelligence for health information: capturing concepts and trends in social media via query expansion. http://www.ncbi.nlm.nih.gov/pubmed/21893923?dopt=Abstract.

[Tobias, 2011] Tobias, E. (2011). Using twitter and other social media platforms to provide situational awareness during an incident. http://www.ncbi.nlm.nih.gov/pubmed/22130339?dopt=Abstract.

[Topsy, 2010] Topsy (2010). Cool tool: Topsy finds most influential tweeters on any topic. http://gigaom.com/2010/07/15/cool-tool-topsy-finds-most-influential-tweeters-on-any-topic/.

[Topsy, 2014] Topsy (2014). Topsy website. http://topsy.com/.

[Twitter, 2014a] Twitter (2014a). Manhattan, our real-time, multi-tenant distributed database for twitter scale. https://blog.twitter.com/2014/manhattan-our-real-time-multi-tenant-distributed-database-for-twitter-scale.

[Twitter, 2014b] Twitter (2014b). Twitter json documentations. https://dev.twitter.com/docs.

[US Dept. pf Health & Human Services, 2012] US Dept. pf Health & Human Services (2012). A partnership between the public and the government to solve important challenges. https://challenge.gov/?q=334-now-trending-health-in-my-community.

[Vance et al., 2009] Vance, K., Howe, W., and Dellavalle, R. (2009). Social internet sites as a source of public health information. http://www.ncbi.nlm.nih.gov/pubmed/19254656?dopt=Abstract.

[Williams et al., 2013a] Williams, S. A., Terras, M., and Warwick, C. (2013a). How twitter is studied in the medical professions: A classification of twitter papers indexed in pubmed. http://www.medicine20.com/2013/2/e2/. Medicine 2.0: Social Media, Mobile Apps, and Internet/Web 2.0 in Health, Medicine and Biomedical Research.

[Williams et al., 2013b] Williams, S. A., Terras, M. M., and Warwick, C. (2013b). What do people study when they study twitter? classifying twitter related academic papers. https://www.emeraldinsight.com/journals.htm?articleid=17088387.

[Wired, 2014] Wired (2014). This is what you build to juggle 6,000 tweets a second. http://www.wired.com/2014/04/twitter-manhattan/.