Healthcare Twitter Analysis

George Fisher

July 27, 2014

Abstract

'Healthcare Twitter Analysis' is an Open Source project which intends to investigate ways to improve the quality of medical care with Data Science techniques applied to Twitter. This paper is one submission by a participant. The project website is [Mehta and Saama Technologies, 2013]; the GitHub repository for this paper can be found at [Fisher, 2014].

Contents

Part I

Summary

This is just a status update, the project is still in its initial stages.

- **Prior Status**
  It was clear upon joining the project that the data provided would have to be augmented, both by the data from the original tweets and from data elsewhere. At this point, I have completed the task of augmenting the files provided with the data from Twitter, plus geo data, and I am considering what other data would be helpful.
- **Current Status**
  Prior to building their own NoSQL database, Twitter used MySQL. Comparing MySQL and MongoDB for this project is a worthwhile research effort.

  I have done some simple Exploratory Data Analyses. I am sorry to say that while these analyses produced some interesting pictures, I do not see any obvious connection to improving medical research. More data and different analytical techniques seem to be in order.

Part II

Data Acquisition and Management

1 Collect Twitter Data

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

There are 897 csv files provided by Topsy, a Twitter aggregator [Topsy, 2014], somewhere in the neighborhood of 2.5 million tweets [Google Drive, 2014]. The files fairly comprehensively cover tweets concerning a wide range of medical conditions, for a six-month period, however the data included only the text of the tweet, its originating user and a score calculated by Topsy.

While the text might be sufficient for a basic textual analysis, the other data provided by Twitter is clearly of value for more extensive analyses, even as simple as filtering by retweet count or plotting geographic incidence.

My GitHub repo for this project [Fisher, 2014] contains a python program that performs two basic tasks:

1. For each tweet in the Topsy data, requests the full json from Twitter
2. For each record it adds
   - All of the data from the Topsy files
   - Location data, including latitude and longitude from Mapquest [Mapquest, 2014]

The data is processed in batches of 100 tweets, a limit imposed by Twitter for automated requests, and a text file is appended and saved after each batch is processed.

The additional json fields included are listed in an appendix on page 17.

Fewer than 1% of the tweets in the project files were rejected by Twitter (based on their id) and over 75% had sufficient location information to allow latitude and longitude tagging.

The process of running the program is fairly fast: on a Dell Windows 7 laptop it processes about 1,700 tweets per minute; however, the elapsed time is much longer than that because Twitter imposes a limit of around 14,000 tweets per 15-minute interval, so the program goes to sleep every 13,500. Based on timings with my equipment, processing the whole dataset would take somewhere in the neighborhood of 2,500,000/13,500*23 = 4259 minutes/71 hours/3 days, which may sound laughable at first but would really not be a gigantic task. I am considering porting the data to Amazon Web Services (AWS) [Amazon, 2014] for a number of reasons and this could be the first.

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 in general I became convinced that json was more appropriate for two reasons:

1. Every book and paper I have read and every Twitter-analytic program refers to the Twitter data in its json form
2. While MongoDB [MongoDB, 2014] supports csv files, including a utility for csv loading, it is clear that MongoDB's native document structure is that of json and since MongoDB seems like a very useful way to store and access Twitter data, it being the one chosen by most other researchers, storing the data in json format seemed to make the most sense.

2 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]. Despite the shift by Twitter away from MySQL, this project might consider its use since it will not face Twitter's volume or locations problems.

MySQL stores json in a single long text field, a blob, but UDFs are available which parse the json in such a database [MySQL Connect, 2013, StackOverflow, 2011, Flite, 2014], so the database does not have to adapt to complex and changing json structures.

Despite its popularity, NoSQL is not specifically geared to data analysis. As an example of the non-analytic cast of NoSQL, MongoDB specifically and NoSQL databases in general do not support the join operator because of the order-polynomial cost of the cross product operation. MongoDB has Map/Reduce operations built in that are intended to manage aggregations of all sorts but because of the size of the anticipated datasets, the expectation is that the data that finally lands in the database is not going to change much or be aggregated at all.

Storing json as a blob on MySQL would seem to eliminate the possibility of joins there, also. It may turn out that parsing a MySQL blob is equivalent to using MongoDB without MongoDB's efficiencies, and it's hard to see how MySQL could replicate MongoDB's indexing capabilities, but further research would be useful.

I have MySQL and MongoDB on my machine. I know how to use basic MySQL; it's on my to-do list to learn MongoDB well enough to have an informed opinion about it its use. I will also try to use the json UDFs for MySQL to see how that goes.

Somebody might want to research other NoSQL systems: I, for one, have no idea whether CouchDB is better than MongoDB for this project [StackOverflow, 2012, Hurst, 2010, Cattell, 2013]; Andrei Krishkevich mentioned Neo4j, which is another one I have no idea about [Krishkevich, 2014].

3 Supplemental Data

Tim Cook [Cook, 2014] has begun work on adding ontology data from BioPortal [BioPortal, 2014]; others have talked of adding physician and hospital data.

Finding additional data for the tweets to allow more extensive analyses is clearly a very important area of research in the near term.

Part III

Exploratory Data Analysis

4 Sentiment

4.1 Summary of Sentiment Analyses

While the pictures are very pleasing, it is not at all clear to me how rudimentary sentiment analysis will provide any value to medical researchers; presumably there are deeper, more sophisticated techniques that provide more useful insights.

4.2 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**

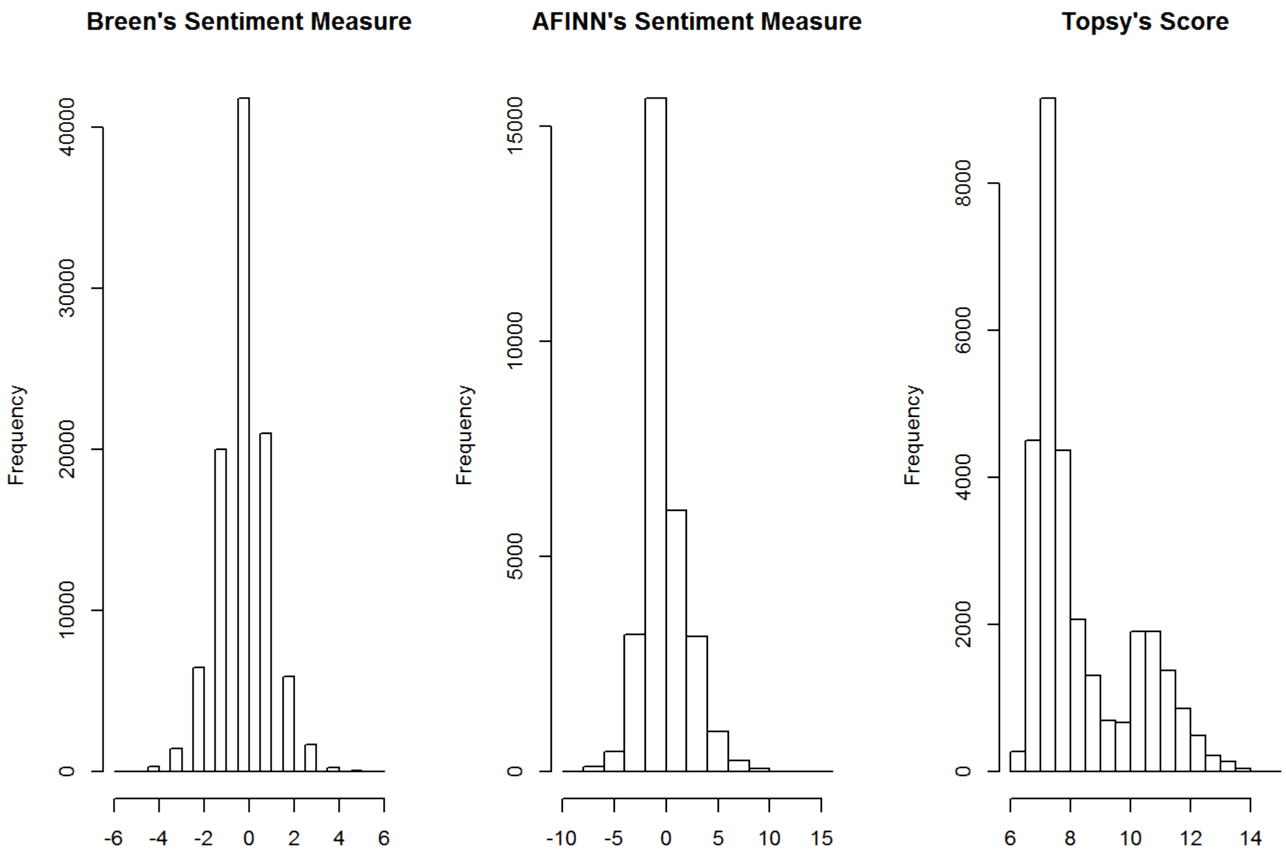


Figure 1: **Distribution of Sentiment Measures and Score** Breen and AFINN are more similar to each other than to score: both have a mean of nearly zero and both are symmetrical around it; but AFINN has a much greater variance and non-normal tail behavior. Score has more of a log or Poisson shape to its distribution, which is bimodal, and is clearly different from the two sentiment scores.

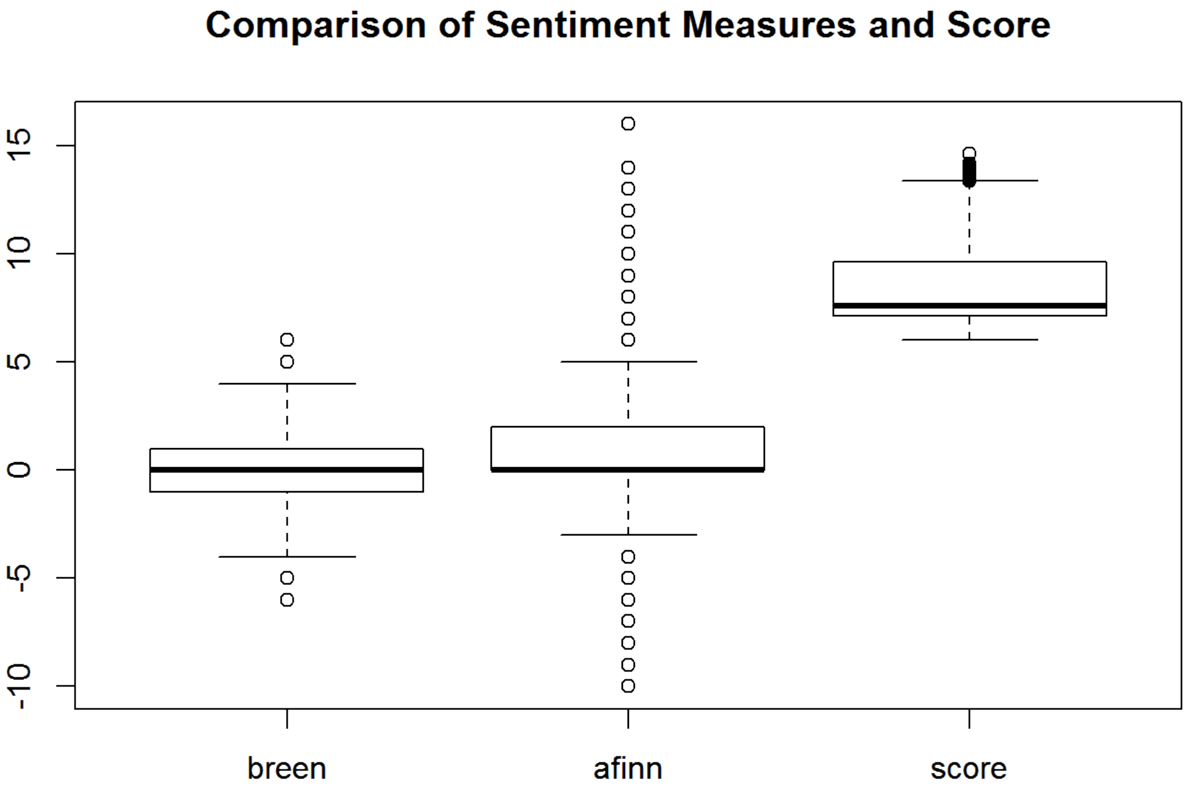


Figure 2: **Distribution of Sentiment Measures and Score** A box plot shows even more starkly the difference in the distributions of these three measures.

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.

4.3 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

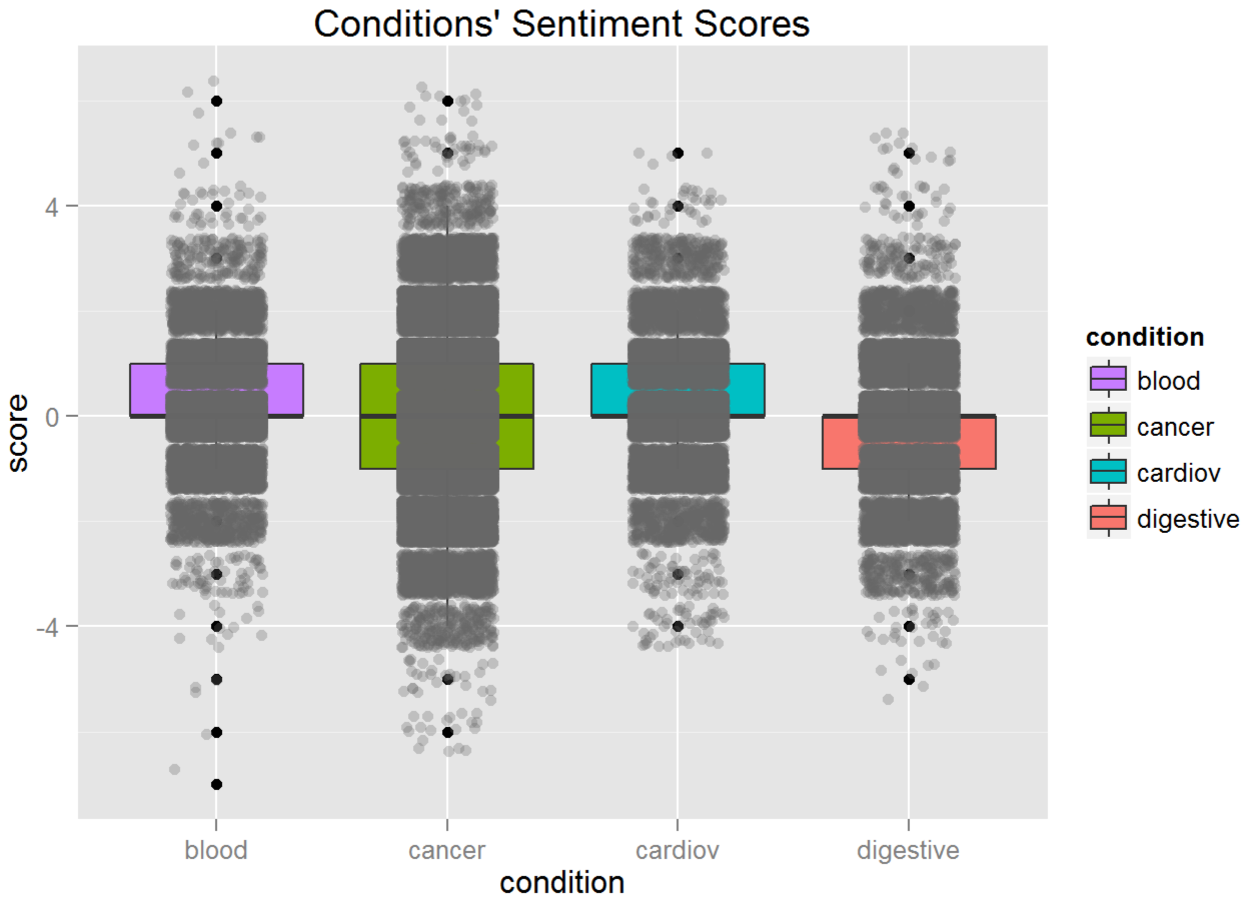


Figure 3: **Distribution of sentiment: Boxplots** The dark gray dots represent the individual data points, roughly 14,000 per condition. The boxes in color represent the inter-quartile distribution of the sentiment for each condition, with bold dots above and below representing outliers beyond the inter-quartile 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.

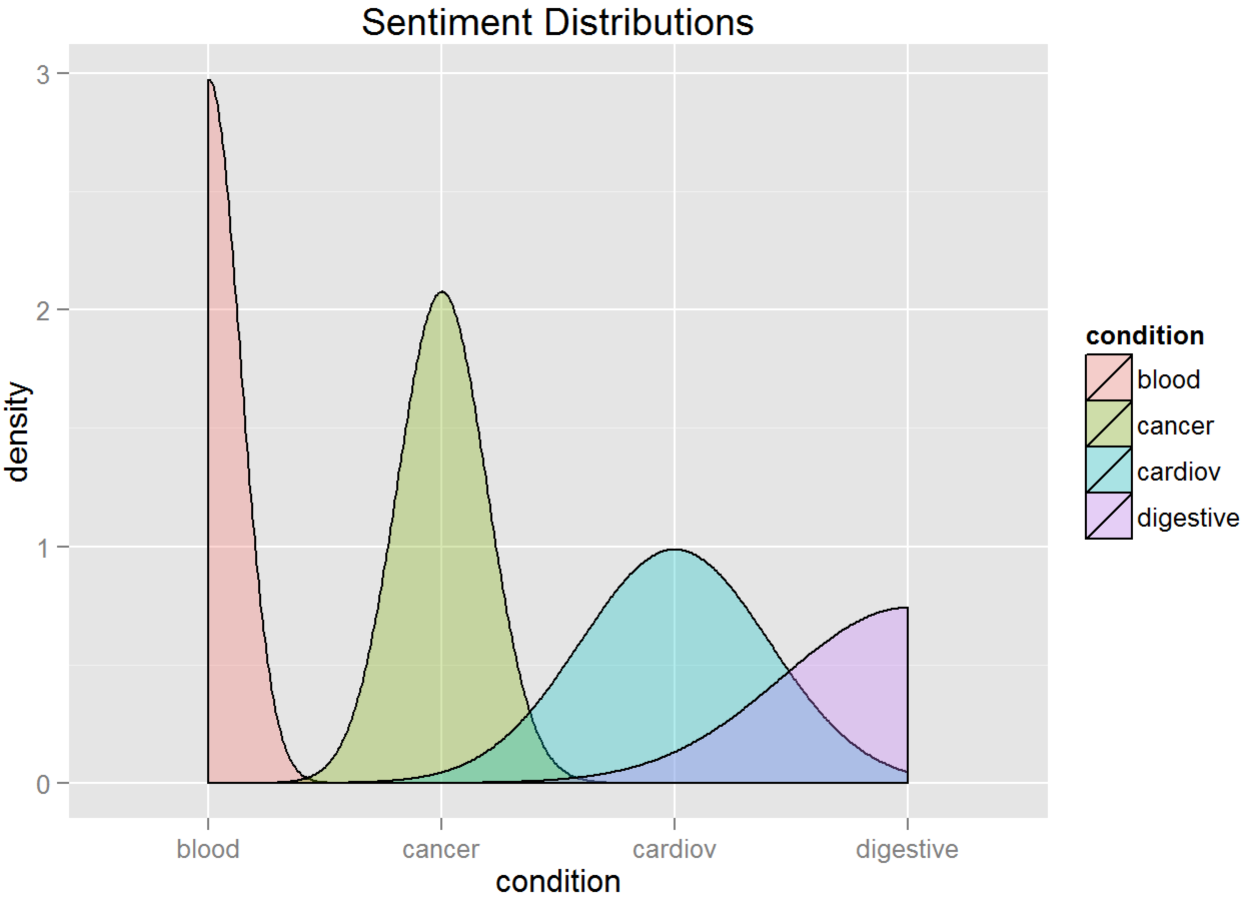


Figure 4: **Distribution of sentiment: Histograms** Another way to look at the distribution of sentiment is to show a smoothed histogram. For each condition, the vertical white line over the label is plotted over the average for that category and the plot shows the distribution around the mean although 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 4.2 beginning on page 6, 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.

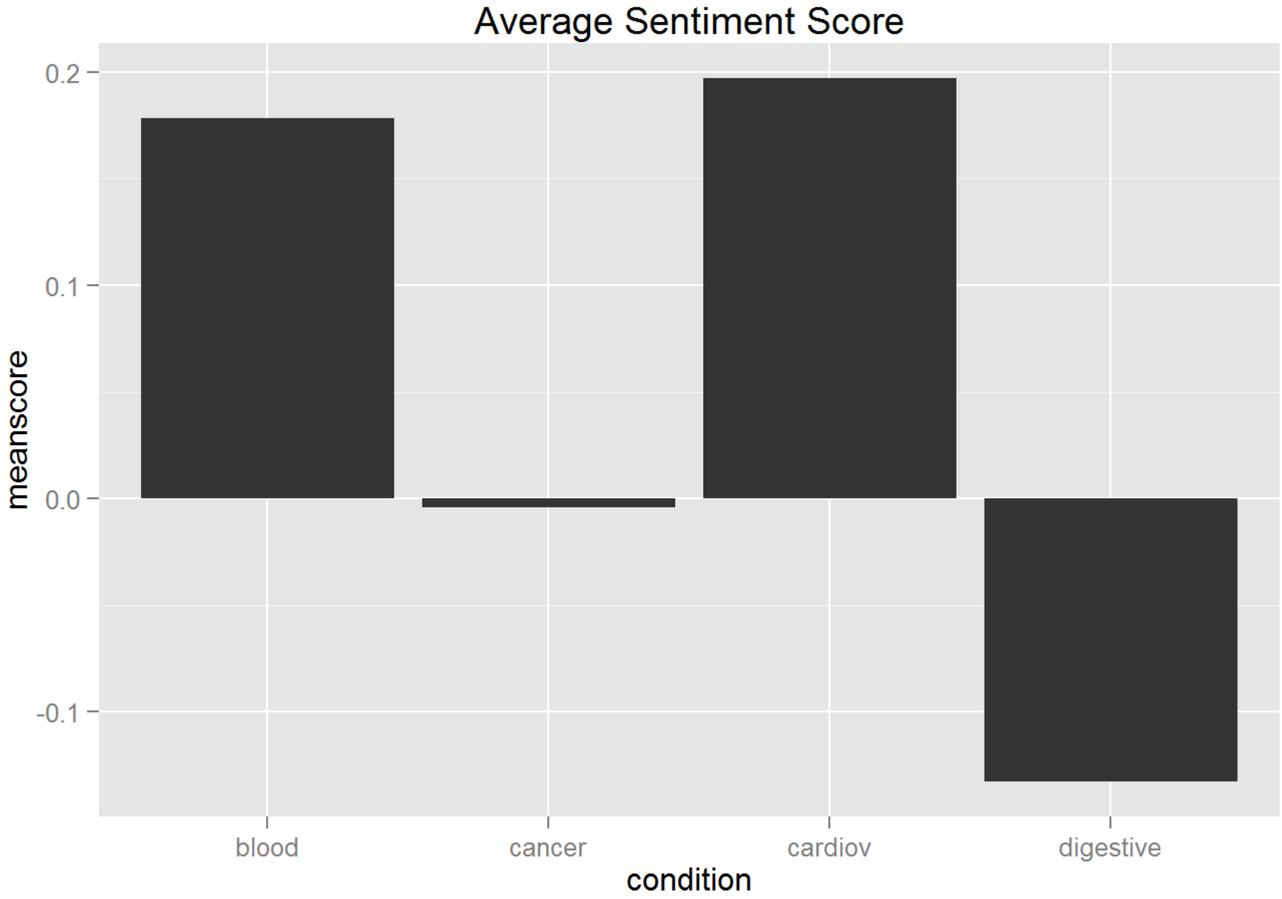


Figure 5: **Average Scores** The averages show us very starkly what we saw in the distributions: Digestive disorders seem to have by far the most negative effect on their suffers and/or those who tweet about them.

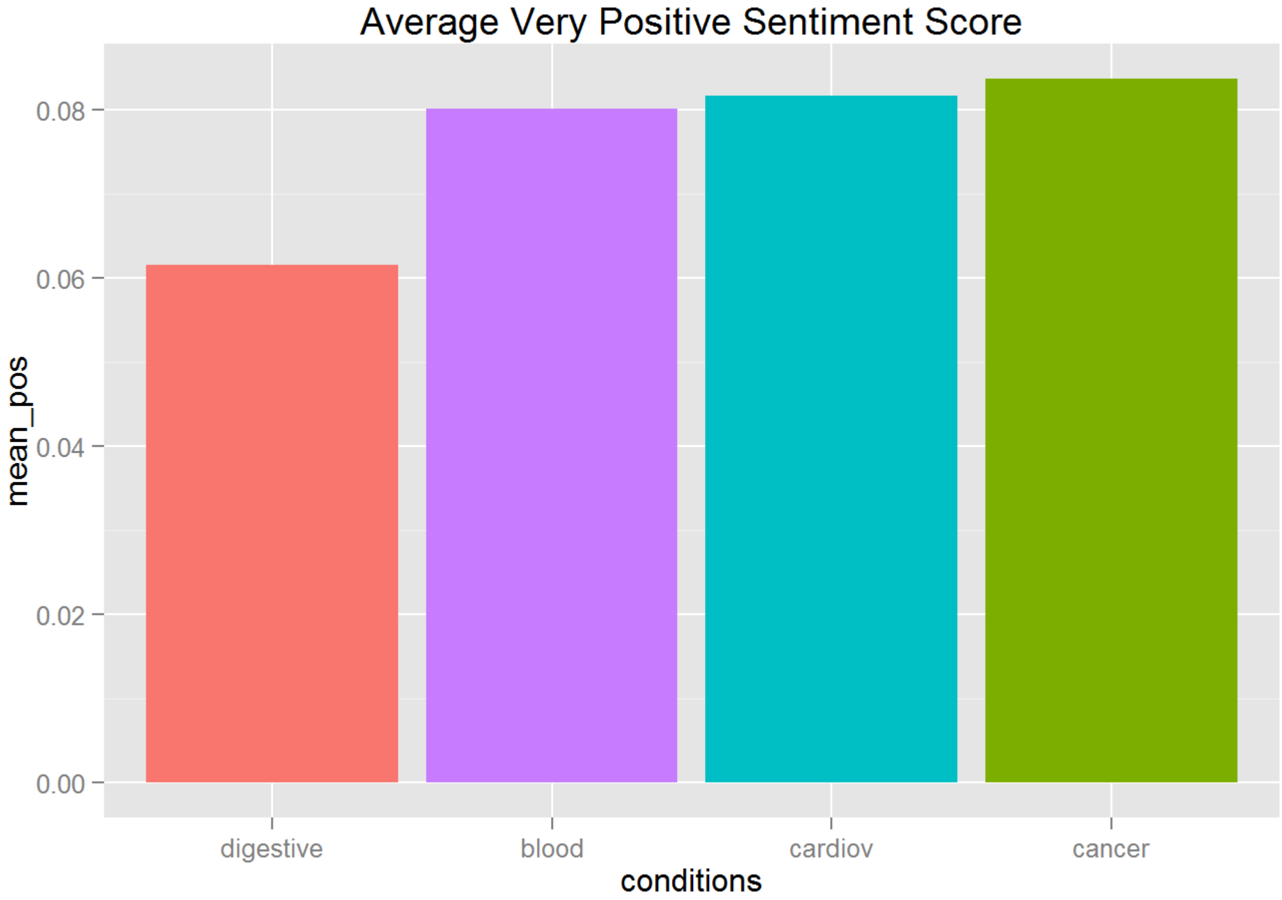


Figure 6: **Average Positive Scores** Looking at the mean scores for only those with a positive sentiment provides more reinforcement for what we have already seen: digestive disorders have a negative psychological effect to the extent of having the lowest mean positive scores.

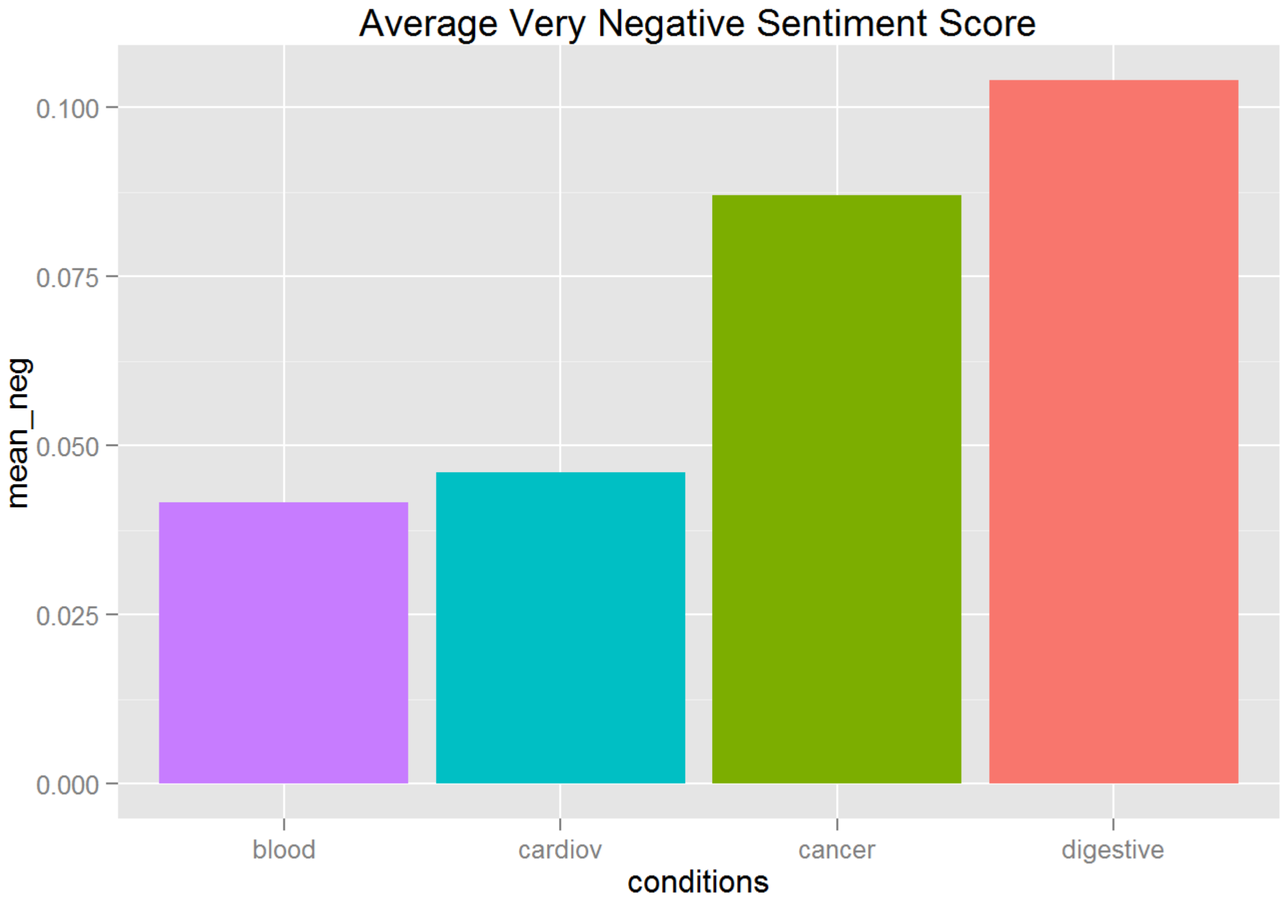


Figure 7: **Average Negative Scores** Looking at the mean scores for only those with a negative sentiment tells the same story: none are good, but of these four, tweets about Digestive Disorders show the greatest tendency toward negativity.

5 Analysis of Text

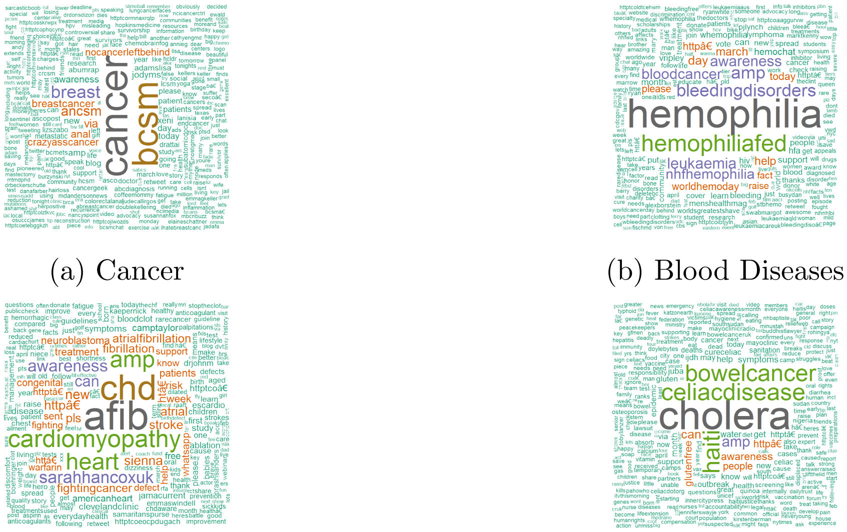
5.1 Summary

Similar to my observation about Sentiment Analysis, I find the pictures for the simple, common textual analyses interesting but I do not see a connection to helping medical research. If value is to be added in this area it will have to come either from the inclusion of additional tags found outside the tweets themselves or else from more sophisticated techniques; perhaps both.

5.2 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 1 starting on page 3.

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 8: Simple Word Clouds for Four Medical Conditions

Cancer
Blood
Cardiov
Digestive
breast
bcsm
hemophilia
cancer
cholera
haiti
chd
afib
celiacdisease
cardiomyopathy
leukaemia
bleedingdisorders
bloodcancer
awareness
amp
can
new
suppo
help
one
treatment
health
spread
take
research

(a) Comparative

(b) Commonality

Figure 9: **Comparison 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.

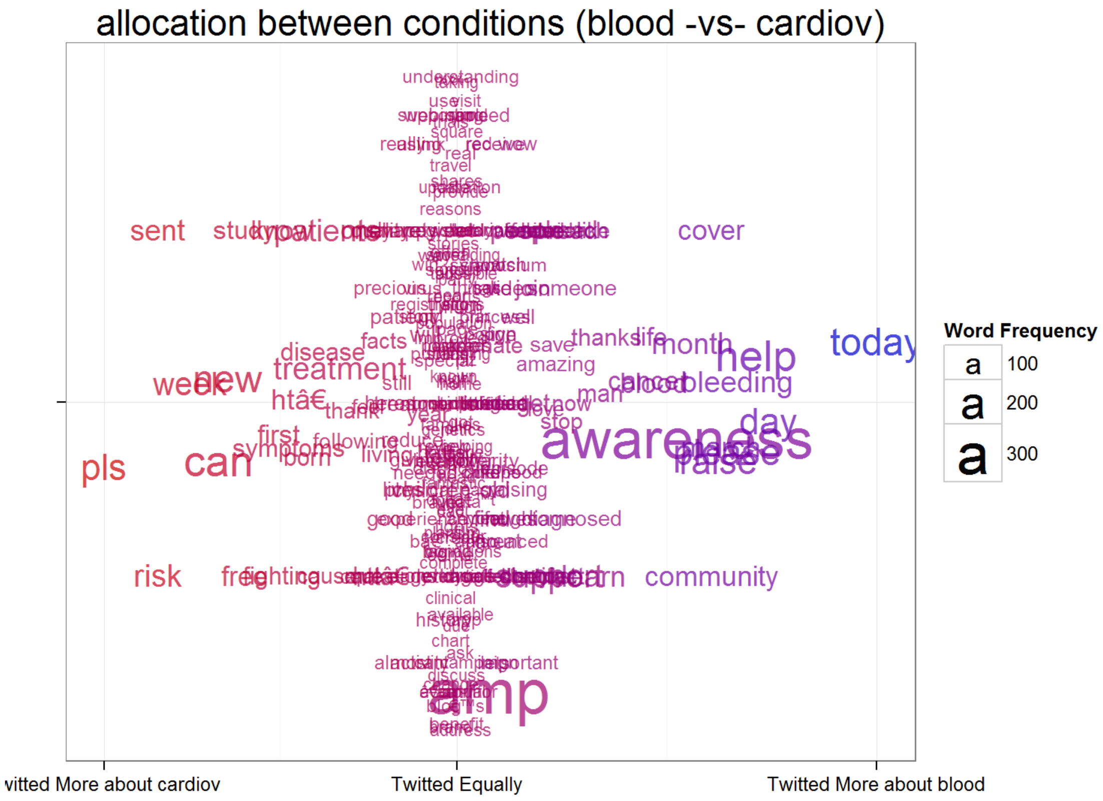


Figure 10: **Conway Comparative Word Cloud of Two Medical Conditions** Comparative word clouds compare all categories together. Conway word clouds show how two categories allocate words between them.

Appendices

fields_added_to_twitter_json.txt

This file shows examples of the json fields added to the full Twitter data by the python program `get_twitter_json.py`, which can be found in the `code` folder on GitHub [Fisher, 2014]. The program's docstring contains examples of using the file this program produces in Python and R as well as loading it into a MongoDB collection. A sample file produced can be found in the `files` folder of the GitHub repo.

The official guide to Twitter's json structures is here: [Twitter, 2014b]; the guide to Mapquest programming is here: [Mapquest, 2014].

The reason for adding the Unix timestamps is for efficient searching in MongoDB; I expect these dates will be part of an eventual index structure and dates in text format are useless for this.

```
get_twitter_json.py augments the Twitter json with the following fields
=======================================================================

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

"user": {

    "location": "Perth, Western Australia",                   # given

    "location_geoinfo": {                                      # derived
            "sideOfStreet": "N",
            "linkId": "282859405",
            "mapUrl": "http://www.mapquestapi.com/staticmap/v4/getmap?key=Fmjtd|luur2008n9,
            "displayLatLng": {
                "lat": -31.952712,
                "lng": 115.86048
            },
            "adminArea6Type": "Neighborhood",
            "adminArea3Type": "State",
            "dragPoint": false,
            "geocodeQualityCode": "A5XAX",
            "adminArea1": "AU",
            "geocodeQuality": "CITY",
            "adminArea3": "Western Australia",
            "adminArea4": "",
            "adminArea5": "Perth",
            "adminArea6": "",
            "postalCode": "",
            "type": "s",
            "adminArea1Type": "Country",
            "adminArea5Type": "City",
            "latLng": {
                "lat": -31.952712,
                "lng": 115.86048
            },
            "adminArea4Type": "County",
            "unknownInput": "",
            "street": ""
        },

"topsy": {

    "firstpost_date": "01/06/14",
    "timestamp": 1388984400.0,         # unix timestamp for "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,                 # info about get_twitter_json.py process
    "short_file_name": "Jan to May\\Blood\\Tweets_BloodCancer.csv"
}
```

References

[Amazon, 2014] Amazon (2014). Amazon web services website. https://aws.amazon.com/.

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

[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/.

[Cattell, 2013] Cattell, R. (2013). Scalable datastores. http://cattell.net/datastores/.

[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.

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

[Flite, 2014] Flite (2014). Faster json parsing using mysql json udfs. http://mechanics.flite.com/blog/2014/02/14/faster-json-parsing-using-mysql-json-udfs/.

[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.

[Hurst, 2010] Hurst, N. (2010). Visual guide to nosql systems. http://blog.nahurst.com/visual-guide-to-nosql-systems.

[Krishkevich, 2014] Krishkevich, A. (2014). Database for n-grams. https://www.coursolve.org/need/184.

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

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

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

[MySQL Connect, 2013] MySQL Connect (2013). Mysql json functions. http://www.slideshare.net/SvetaSmirnova/mysql-json-functions.

[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.

[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.

[StackOverflow, 2011] StackOverflow (2011). Mysql udf for working with json? https://stackoverflow.com/questions/8031878/mysql-udf-for-working-with-json.

[StackOverflow, 2012] StackOverflow (2012). When to use couchdb over mongodb and vice versa. http://stackoverflow.com/questions/12437790/when-to-use-couchdb-over-mongodb-and-vice-versa.

[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.

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