Mining Twitter Using R

Presented to the St. Louis R Users Group
By Mathew Woodyard
April 4, 2012

1 Motivation

We endeavor to answer the questions: Why aren't people buying my widgets? Are people saying bad things about me? Why are the students of Southern Illinois University Edwardsville (SIUE) unhappy?

These are classic questions concerning not only product marketers and statisticians: what convinces people to tell us what they think, what they own, what they will buy, where they are, and who they are? How do we get honest answers? What can we infer from a large number of these answers?

There is some good news yet because...

1.1 300 Million People Are Here To Chat!

Twitter is a "microblogging" service that facilitates small bits of communication. Each message, a "tweet", contains no more than 140 characters. 300 million people use Twitter. But how do they use it? Why? How does this help us?

Twitter has some use cases and features that result in a nice corpus. It is often used for consumption of news stories and participation in current events. Many people also use Twitter to express emotional states: often complaining about or praising places, products, and services. Since tweets contain no more than 140 characters, finding relevant tweets is normally a fairly easy task. Twitter's automatic grouping and "hashtags", tags that group topics and start with a hash mark (#), enable us find what we want with relative ease.

In my example, I will use tweets about SIUE. Tweets were selected from the stream based on the occurrence of the terms "SIUE" and "#onlyatsiue", the latter having spent some time as a major regional trending topic.

Figure 1. A tweet in its natural habitat.

Now that we know people are generating data we can use, how can we provide analysis that makes these tweets useful?

2 Harnessing Tweets

Our strategy will consist of the following steps.

1. Sip from the stream of tweets using twitteR and ROAuth to build a corpus.
2. Discover our new corpus.
3. Select a sentiment lexicon. Download it and load it into R.
4. Score tweets using the function provided or using your own classification scheme.
5. Summarize the results.

2.1 Building a Corpus

First, we sip from the stream to build a corpus. Be aware that we can only sip, and not guzzle, due to Twitter's rate limiting (note there is no limit on the number of tweets you post using twitteR). The code below loads some previously-downloaded tweets. Code to append new tweets from the stream is commented out and can be run when you want to expand the corpus. Note that by registering with Twitter's developer program, you can get API keys used to increase the rate limit for your IP/key combination.

```
> # Load needed libraries to allow R to interface with twitter and
> # interface with the API using OAuth.
> require(twitteR)
> require(ROAuth)
> # The code below does not work due to a current bug in twitteR.
> # This means that the rate of downloading tweets will be limited.
> # The following code should NEVER be shared. It includes private keys
> # that authenticate against the twitter API. This code was taken from the
> # twitteR documentation. Run the code from this block to the next comment.
> #
> # reqURL <- "https://api.twitter.com/oauth/request_token"
> # accessURL <- "https://api.twitter.com/oauth/access_token"
> # authURL <- "https://api.twitter.com/oauth/authorize"
> # consumerKey <- "keygoeshere"
> # consumerSecret <- "keygoeshere"
> # twitCred <- OAuthFactory$new(consumerKey=consumer.key,
> #                              consumerSecret=consumer.secret,
> #                              requestURL=reqURL,
> #                              accessURL=accessURL,
```

```
> #                                 authURL=authURL)
> # twitCred$handshake()
> #
> # After entering the PIN provided by twitter, run the following command.
> #
> # registerTwitterOAuth(twitCred)
>
> # Load saved tweets.
> load(file="../../siue.tweets.RData")
>
> # Uncomment to add hot new tweets.
> # siueTweets <- append(siue.tweets, searchTwitter('siue', n=1500))
> # siueTweets <- append(siue.tweets, searchTwitter('#onlyatsiue', n=1500))
```

2.2 Exploring Your Hot New Corpus

Now that we have the corpus loaded, let's explore what people from SIUE are saying about their university.

```
> tail(siue.tweets, 2)

[[1]]
[1] "ktdare: Fire alarm? Check. Short power outage? Check. #onlyatsiue"

[[2]]
[1] "ChrisHeinle2211: thought a goose was going to attack me #OnlyAtSiue"
```

So far we have some anecdotes and can probably infer that their university had a power outage, is a goose sanctuary, and has a bit of a wasp problem. But what kind of sentiments are associated with SIUE in the aggregate?

2.3 Loading Lexicons; Tweaking Tweets

Before we can preform more analysis on the aggregate sentiments within these tweets, we need to first download and load a sentiment lexicon into R. Then we need to modify the corpus so that we can analyze it.

The Sentiment Lexicon for this project is taken from Hu and Liu's opinion lexicon. See `http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html` for source information. You can find download their opinion lexicon from `http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar`. The raw text is also included in the GitHub repository for this project.

```
> # Import sentiment lexicons. If the situation demands,
> # add in domain-specific words.
> positiveWords <-
```

```
+   scan('../../Hu and Liu Sentiment Lexicon/positive-words.txt',
+        what='character', comment.char=';')
> negativeWords <-
+   scan('../../Hu and Liu Sentiment Lexicon/negative-words.txt',
+        what='character', comment.char=';')
```

Now we need to restructure the corpus so we can preform some analysis.

```
> # Extract the text of the tweet for mining.
> tweetText <-lapply(siue.tweets, function(x) x$getText())
> # Remove non-UTF8 tweets.
> tweetText <- enc2utf8(gsub("[\U80-\UFF]", "", tweetText, useBytes=TRUE))
```

Great. The tweets are ready to be scored.

2.4 Scoring Tweets

The function we are using to score tweets is mainly taken from “R by example: mining Twitter for consumer attitudes towards airlines” by Jeffrey Breen. If you want to see the function, simply look at the R code in the GitHub project or in the Rnw file that comes with this PDF.

```
> tweetScores <- score.sentiment(tweetText, positiveWords, negativeWords)
```

The scoring is complete.

2.5 Summarizing Your Results

Now we summarize the results of our mining with a humble histogram. I have selected `ggplot2` for this task, mainly because it is elegant by default and has a function (`qplot`) that is intuitive to native R graphics users.

```
> require(ggplot2)
> qplot(score, data=tweetScores, geom="histogram", binwidth=1)
```

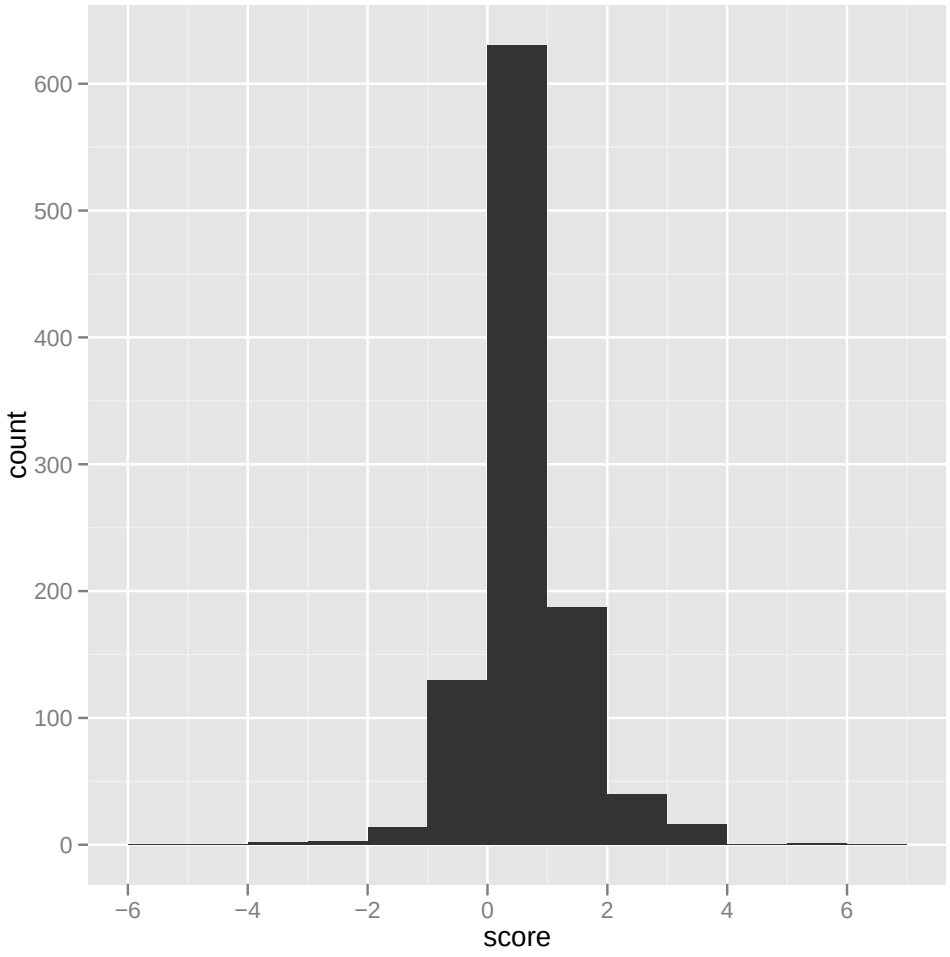

Figure 2. Histogram of tweet scores.

From exploring the data, we know that SIUE has tense animal-human relations and had a recent power outage and fire alarm. Our summary shows that this sample of tweets about SIUE are (very) slightly positive. Combined with input from subject matter experts, further analysis, and insights from other data, we can start to get a picture of what life is like at SIUE.

3 Limitations

Limitations exist in the methodology I have presented. One class of limitations is related to the over-simplified model implicitly used and the other is related to R.

3.1 Limitations of Methodology

We used a bag-of-words model when we analyzed the sentiment of the tweets. In the process we disregarded grammar and word order and ranked the tweets strictly based on the words the tweet contained. There are other ways to process the tweets that give more sophistication to the analysis.

Additionally, the function used to score the tweets is completely deaf to irony. For example, the (fabricated) tweet

> Impressed and amazed by the peerless leadership of SIUE in the domain of mediocrity.

actually receives a score of +2. While all these problems have solutions or techniques to mitigate them, that is beyond the scope of this presentation.

3.2 P*R*oblems

Another source of trouble is R itself. When using R for text mining and text processing, one often feels as if one is reinventing the wheel. Other languages like Python benefit from existing frameworks like NLTK and Beautiful Soup. That being said, there are some packages that make text mining easier in R. `tm` is a good example of a package that makes text mining easier.

4 Credits

To complete this project, I stood on the shoulders of giants (as always the case when using R). We have received aid and comfort from the packages `plyr`, `stringr`, `twitteR`, `ggplot2` and `ROAuth`. For more information on these packages, see their associated help pages and vignettes.

We also used the sentiment lexicons of Minqing Hu and Bing Liu in order to do the scoring for our analysis. In the process of scoring tweets, much inspiration was taken from Jeffrey Breen's presentation. We have all been helped more than we can express by the members of the R community.

5 License