

# Big Data Velocity, or Where and Why is this Data Coming so Fast?

Tidewater Big Data Enthusiasts  
Chuck Cartledge  
Developer

26 April 2016

## Contents

<b>List of Figures</b>	<b>ii</b>
<b>List of Tables</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Discussion</b>	<b>1</b>
2.1 Twitter . . . . .	3
2.2 Tweets . . . . .	3
2.3 Twitter Application Program Interface (API) . . . . .	5
2.4 Sentiment Analysis . . . . .	6
<b>3 Results</b>	<b>9</b>
<b>4 Conclusion</b>	<b>19</b>
<b>A Anatomy of a Tweet</b>	<b>20</b>
<b>B Real-time software</b>	<b>29</b>
<b>C Misc. files</b>	<b>30</b>
<b>D References</b>	<b>31</b>

## List of Figures

1	Doug Laney. . . . .	1
2	Doug Laney's original Big Data 3Vs. . . . .	2
3	Twitter account sign up page. . . . .	4
4	Twitter API sign-up page. . . . .	6
5	Twitter API details page. . . . .	7
6	Twitter API keys page. . . . .	8
7	Real tweet sentiments over time. . . . .	10
8	A sample connection graph started from one tweet. . . . .	12
9	A Pareto graph of connections from the sample graph. . . . .	13
10	A connection graph including all discovered tweeters. . . . .	14
11	A Pareto graph of connections including all discovered tweeters. . . . .	15

## List of Tables

1	Most well connected tweeter of sample tweets. . . . .	16
2	Anatomy of a tweet. . . . .	20
3	Anatomy of a tweet:user. . . . .	23
4	Anatomy of a tweet:entity. . . . .	27
5	Anatomy of a tweet:places. . . . .	27
6	Real-time software and system classifications. . . . .	29

# 1 Introduction

Doug Laney (see Figure 1) originated Big Data's 3Vs: volume, variety, and velocity[3]. In the late 1990s, it was becoming evident to him and his business customers that something was going on in the world of business data. Something that hadn't happened before. Electronic commerce, post Y2K was making it easier for companies to collect data from different sources, the data was in different formats, and the flow data was increasing at an alarming rate.

In an effort to explain what he was seeing to his customers, he created a graphic (see Figure 2 on the next page), to enable them to understand the world as he saw it and how traditional relational database management systems weren't designed for this new environment. Laney's 3Vs of volume, variety, and velocity caught on. Others have added an assortment of Vs to the original ones<sup>1</sup>. Laney has even alluded to expanding to 12Vs[4].

Laney's velocity dimension evolved from the number of people and organizations that had an interest in creating and collecting more and more data. In our Big Data exploration, we do not have the luxury(?) of having lots of companies creating data for our use. But, we can have lots of independent people create data for us that we can use.

Our Big Data velocity exploration will be fueled by using tweets from Twitter. We will walk through the steps to gain relatively unfettered access to tweets, analyse them in soft real-time (see Section B on page 29) for sentiment and categorization, and display the results in different ways (see Section 3 on page 9).



Figure 1: Doug Laney.  
*The man credited with the original Big Data 3-Vs.*

# 2 Discussion

Since Laney popularized the idea of Big Data, his ideas and terms have moved outside the world of Mergers and Acquisitions. His Vs have been recast many times, depending on how and where they are used. In our little world, we will also recast them into:

*"Big Data is a new term used to identify datasets that we can not manage with current methodologies or data mining software tools due to their large size and complexity. Big Data mining is the capability of extracting useful information from these large datasets or streams of data. New mining techniques are necessary due to the volume, variability, and velocity, of such data."*

Fan and Bifet [2]

---

<sup>1</sup>In February 2016, we counted 19 different Big Data Vs before we stopped counting.

**Figure 1 — Data Management Solutions**

▲ **Volume**

- Tiered storage/hub and spoke
- Selective data retention
- Statistical sampling
- Redundancy elimination
- Offload “cold” data
- Outsourcing

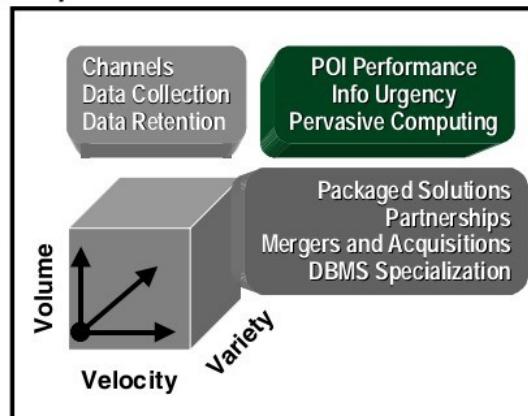
▲ **Velocity**

- Operational data stores
- Data caches
- Point-to-point data routing
- Balance data latency with decision cycles

▲ **Variety**

- Inconsistency resolution
- XML-based “universal” translation
- Application-aware EAI adapters
- Data access middleware and ETL
- Distributed query management
- Metadata management

● **E-Business-Driven Information Explosion Factors**



***Extending data management options enables greater returns on information assets***

Source: META Group

Figure 2: Doug Laney’s original Big Data 3Vs.

Our discussion and demonstration will be a very small sample of tweets. Small enough that they can be processed in a reasonable amount of time on middle of the road home computers. Although we will be working with a small system, the ideas and techniques will scale up (with modest changes) to enterprise sized systems.

## 2.1 Twitter

*“Twitter is an online social networking service that enables users to send and read short 140-character messages called ‘tweets.’ ”*

Wikipedia [13]

Twitter (and its tweets) has become a fast and easy way to share current events, happenings, notifications, and thoughts using a variety of input and output devices. Within the twitterverse, there are two types of people. Those who are registered, and those who are not registered. People who are not registered can search, and monitor tweets. People who are registered can send tweets in to the twitterverse, as well as do all the things that an unregistered person can do.

For our exploration into Big Data velocity, you must be registered with Twitter. Registration is free, and no credit card information is required (see Figure 3 on the following page).

## 2.2 Tweets

*“A Tweet is any message posted to Twitter which may contain photos, videos, links and up to 140 characters of text.”*

Twitter Staff [10]

At its roots, a tweet is an extension of Global System for Mobile Communications (GSM) developed in the early 1990s to exchange short messages (160 characters) between mobile users. Over time, the name changed from GSM to short messaging service (SSM) [15]. Tweets were limited to 140 characters to standardize the number of characters that each user could send. (Originally, sender identification was included in the number of characters a sender could send. So users with short IDs could send more information in the text compared to users with longer IDs.)

While the number of characters is limited, there are many innovative ways to maximize the use of those characters. One that is used widely is the URL shortening service bitly.<sup>2</sup> bitly shortens a uniform resource locator (URL) to 16 characters, regardless of how many characters are in the URL[9]. This shortened URL can then be sent via a tweet. Behind the

---

<sup>2</sup><https://bitly.com/>

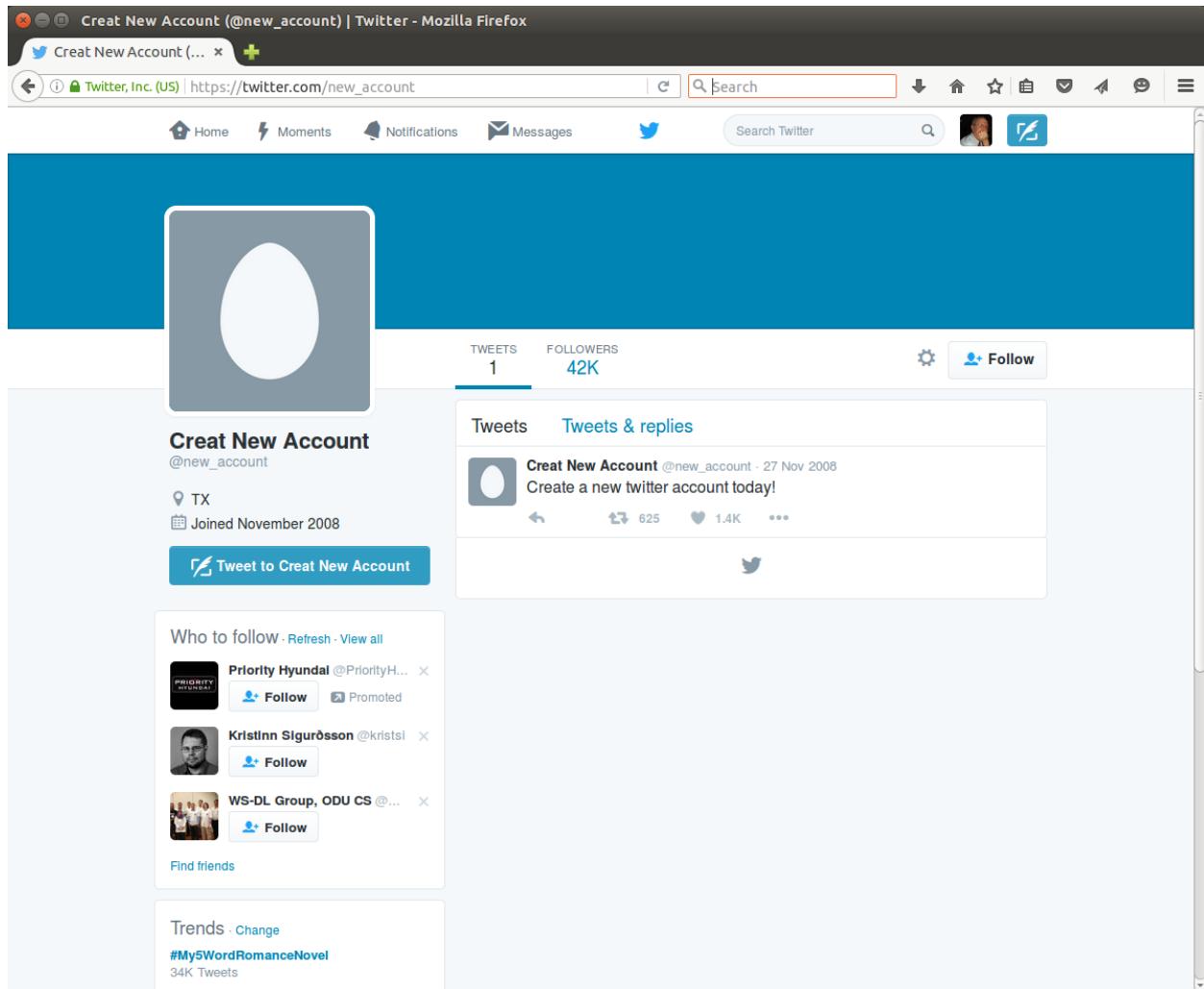


Figure 3: Twitter account sign up page. *The initial signin screen ([https://twitter.com/new\\_account](https://twitter.com/new_account)). A valid twitter account is required to search the twitterverse, either through a UI or an API.*

scenes, when the recipient clicks on the shortened URL, bitly receives the click, retrieves the full length URL from its database, and redirects the clicker to the original URL.

In the twitterverse, the relationship between people started out as “friends” and “followers.” Person A identified as a “follower” of person B, and would therefore get a copy of all notifications from person B. This means that A was “following” B. Persons C and D were identified as “friends” if C followed D and D followed C. Friends had a bidirectional relationship, whereas followers had a unidirectional relationship. These terms were confusing to users, so were gradually replaced with the idea of “notifications” [14]. While the ideas of friends and followers have been disappearing from most twitter applications and user interfaces, they still exist in the background and figure prominently in the application program interface (API) arena.

### 2.3 Twitter Application Program Interface (API)

To access tweets with a custom program, you have to become a Twitter developer. Becoming a developer does not obligate you to make your program public, nor to pay Twitter to access their tweets. In essence, they will allow you to access their data, but will restrict how often you can access the data. If you want unbridled access to their data, then you can pay them for that access. In our exploration, we will take the free route.

The twitter API sign-up page requires only a few pieces of information (see Figure 4 on the next page). They are:

1. Name: what is the name of this application. This name has to be unique to the account used to create application.
2. Description: just a short description of what the application will do.
3. Website: a correctly formed URL. The URL does not have to be functional.
4. (Optional) Callback URL: depending on how your software authenticates itself to Twitter, you may have to fill this in. If you do and you are using the R httr package, the recommended value is:

`http://127.0.0.1:1410`

The httr package creates a webserver that listens for on port 1410 of the local machine<sup>3</sup> for Open Authentication (OAuth). Be sure to use the IP address 127.0.0.1 and not the common name `localhost`.

---

<sup>3</sup><https://github.com/hadley/httr/blob/master/R/oauth-listener.r>

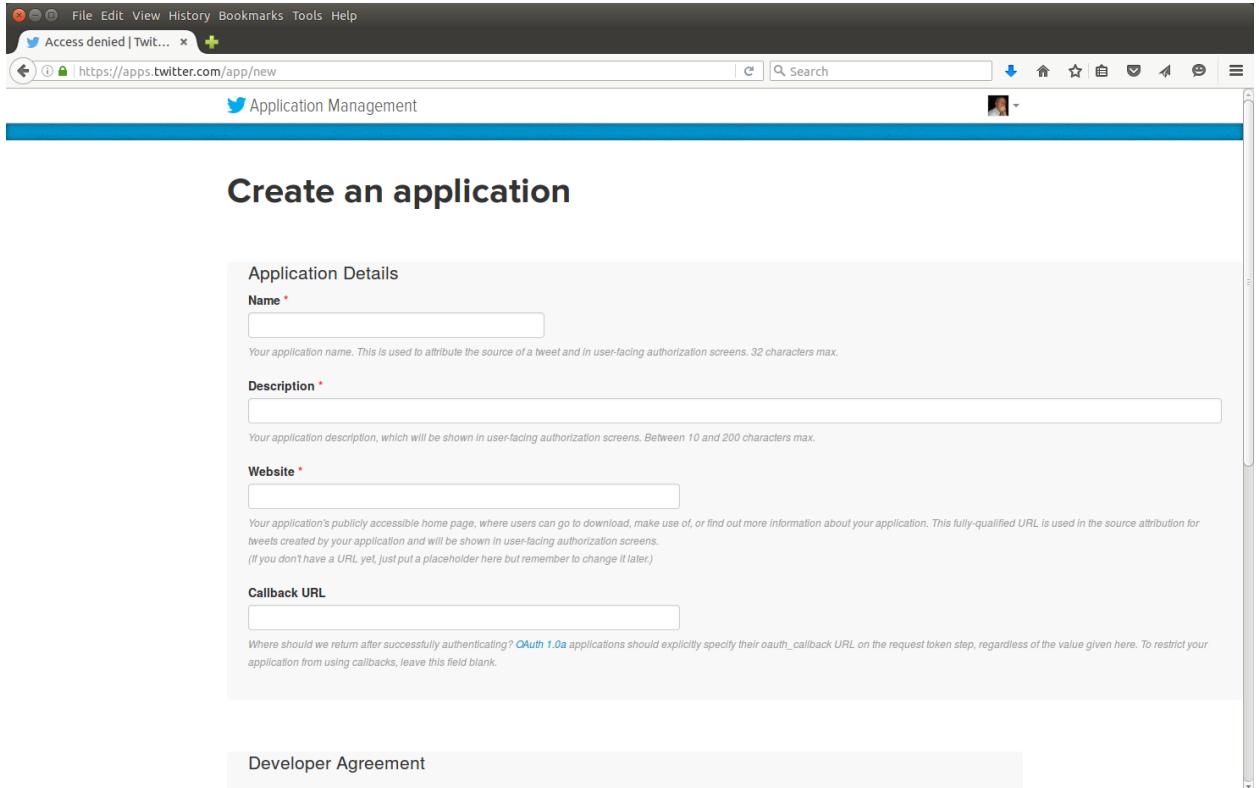


Figure 4: Twitter API sign-up page. *You can sign up for a free twitter development account, once you have a twitter account.*

Once the information is filled out, and the you accept the developer agreement, you will be presented with the application details page (see Figure 5 on the following page). The “Keys and Access Tokens” tabs contain the keys required by many different application packages to use the API. Keys specific to the example application have been redacted.

By this time, you will have defined an application to access tweets, and received the necessary keys to allow your program into the twitterverse. Later, we’ll search for specific words in tweets, and then see how the friends and follower relationships propagate notifications through out the twitterverse.

## 2.4 Sentiment Analysis

Sentiment analysis (sometimes called opinion mining) is applying a computer algorithm to a body of text with the goal of categorizing the text as either positive or negative (or: pro or con, for or against, etc.). These algorithms generally fall into one of three categories[6]:

1. Knowledge-based techniques: categorizing the text based on the presence of unambiguous positive or negative words,

The screenshot shows a web browser window with the title bar "File Edit View History Bookmarks Tools Help" and a tab labeled "Visualize My Tweets...". The address bar shows the URL "https://apps.twitter.com/app/9175807". The main content area displays the "Visualize My Tweets CLC" application settings.

**Organization**

Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

**Application Settings**

Your application's Consumer Key and Secret are used to **authenticate** requests to the Twitter Platform.

Access level	Read and write ( <a href="#">modify app permissions</a> )
Consumer Key (API Key)	[REDACTED] ( <a href="#">manage keys and access tokens</a> )
Callback URL	http://127.0.0.1:1410/
Callback URL Locked	No
Sign in with Twitter	Yes
App-only authentication	<a href="https://api.twitter.com/oauth2/token">https://api.twitter.com/oauth2/token</a>
Request token URL	<a href="https://api.twitter.com/oauth/request_token">https://api.twitter.com/oauth/request_token</a>
Authorize URL	<a href="https://api.twitter.com/oauth/authorize">https://api.twitter.com/oauth/authorize</a>
Access token URL	<a href="https://api.twitter.com/oauth/access_token">https://api.twitter.com/oauth/access_token</a>

Figure 5: Twitter API details page. *Information specific to this account has been blocked out.*

The screenshot shows a Mozilla Firefox browser window with the title bar "Visualize My Tweets CLC | Twitter Application Management - Mozilla Firefox". The address bar contains the URL "https://apps.twitter.com/app/9175807/keys". The main content area is titled "Visualize My Tweets CLC". It features a navigation bar with tabs: Details, Settings, Keys and Access Tokens (which is selected), and Permissions. A "Test OAuth" button is located in the top right corner of this bar. Below the tabs, there is a section titled "Application Settings" with the following details:

Consumer Key (API Key)	[REDACTED]
Consumer Secret (API Secret)	[REDACTED]
Access Level	Read and write ( <a href="#">modify app permissions</a> )
Owner	SimpleSimon2013
Owner ID	1646059915

Below the settings is a "Application Actions" section containing two buttons: "Regenerate Consumer Key and Secret" and "Change App Permissions". At the bottom of the page, under "Your Access Token", it says: "This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone." An "Access Token" field is shown, with its value [REDACTED] followed by a long string of characters. The URL "https://apps.twitter.com/app/9175807/keys" is also visible at the bottom.

Figure 6: Twitter API keys page. *Information specific to this account has been blocked out.*

2. Statistical methods: machine learning techniques using a training body of text to develop models that are then applied to raw text,
3. Hybrid approaches: a combination of knowledge-based and statistical methods.

We will be using a knowledge-based technique described in [1]. The essence of the technique is:

1. Receive a tweet based on user provided search word.
2. “Normalize” the tweet by:
  - (a) Removing all non-ASCII characters
  - (b) Removing Stop Words. Stop words usually refer to the most common words in a language, there is no single universal list of stop words used by all natural language processing tools, and indeed not all tools even use such a list[12, 8].
  - (c) Changing all words to lower case
3. Counting all the text words that are in the list of positive words
4. Counting all the text words that are in the list of negative words
5. Compute the percentage of positive words in the text
6. Compute the percentage of negative words in the text
7. Categorize the text as positive, negative, or neutral based on the text being more than 50% positive or negative. If neither positive nor negative, then the text is neutral.

### 3 Results

We will be applying algorithm from the Discussion section to tweets with the following search words:

- realdonaldtrump
- hillaryclinton
- berniesanders
- tedcruz

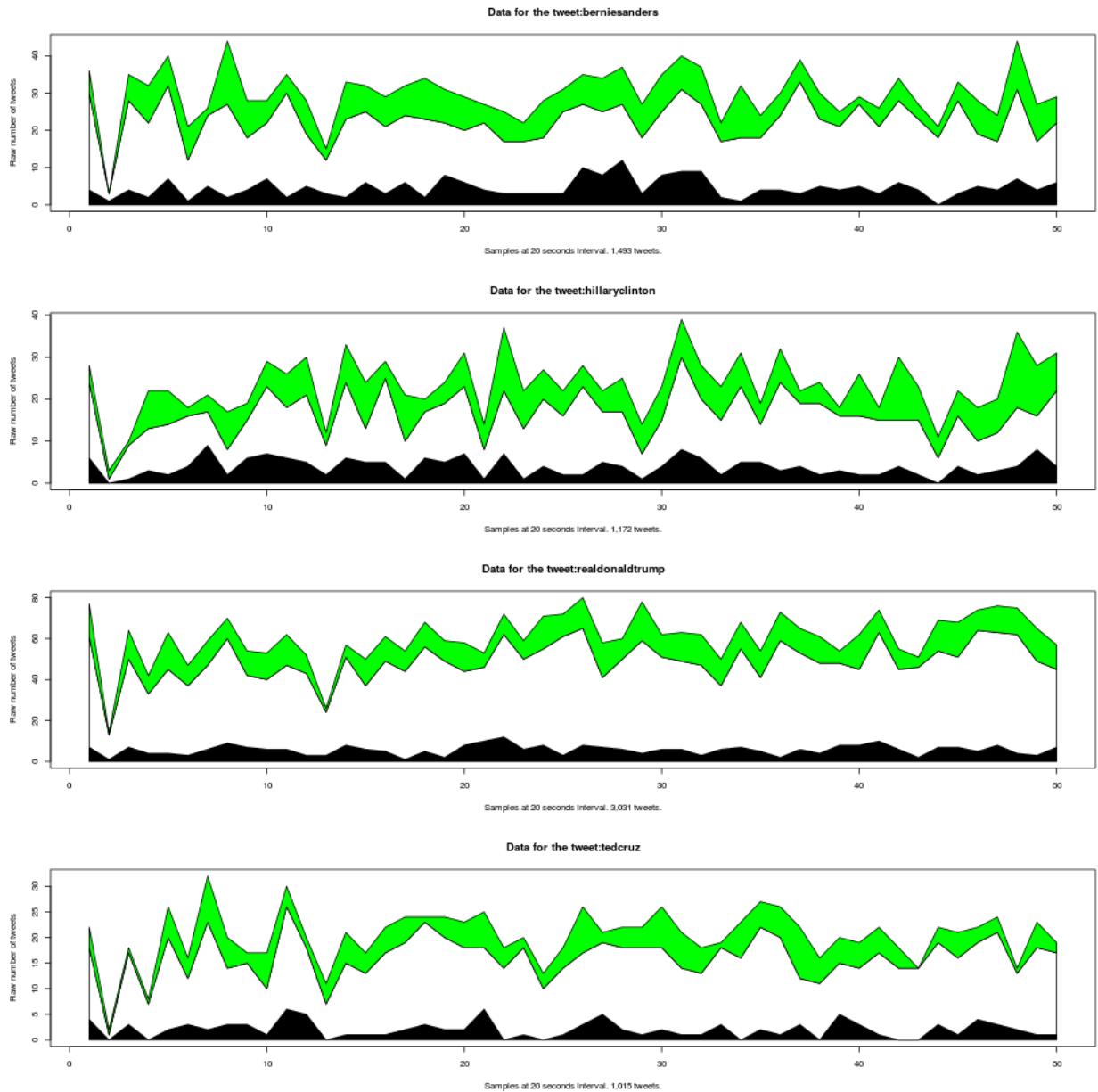


Figure 7: Real tweet sentiments over time. *The green area represent positive tweets, black negative tweets, and white neutral tweets.*

This set of words was chosen under the assumption that a reasonable number of positive and negative tweets from a variety of sources could be collected in a reasonable time. No particular political leaning should be inferred. The tweets were collected for a number of 20 second intervals, and the percentage of positive, negative, and neutral tweets plotted (see Figure 7 on the previous page).

Another way to visualize how tweets are distributed is by looking at the connections made by the old “friend” and “follower” connections. While these designations have been replaced by notifications, they are still evident in the twitter data. Examining the notification connections between tweeters can give insight into who is influential, and who is a listener.

Given any starting tweet, we can see the originator, the number of friends and followers. By requesting information about the originator, we can get a list of friends and followers. Using these connections, we can build a graph that starts with the originator, shows a unidirectional connection to its followers, and bidirectional connections to its friends. We can repeat this process with each of the new connections, identifying friends and followers, until we get tired of tracking down information from Twitter. In a graph-centric environment, these connections are called nodes, and the way they are connected are called edges. We can determine how well connected a node is by looking at the number of edges that are incident to the node.

When we start looking at how nodes (nee, tweeters) are connected (see Figure 8 on the following page), we start to see that some nodes are very well connected (as in they have lots of friends and followers), while others are less well connected. Another way to look at how many nodes are well connected is to create a Pareto chart<sup>4</sup> of the number of edges that each node has (see Figure 9 on page 13). From this we can see that the vast majority of the nodes have very few connections (that we have discovered so far), and that a few have a very large number. These ideas can be extended to the total Twitter graph (as far as we have discovered it) (see Figures 10 on page 14 and 11 on page 15).

Yet another way to look at the most connected tweeters, is by name and personal description (see Section A on page 20). Tweets were collected using the search term “data mining” to create a different data set. A user name can be indicative of what a person thinks about themselves (see Table 1 on page 16).

---

<sup>4</sup>A Pareto chart is a specialized histogram where the data on the horizontal axis is ordered by the frequency of the data being plotted. A histogram is a diagram consisting of rectangles whose area is proportional to the frequency of a variable and whose width is equal to the class interval.

**9669 unique nodes, 16906 edges**

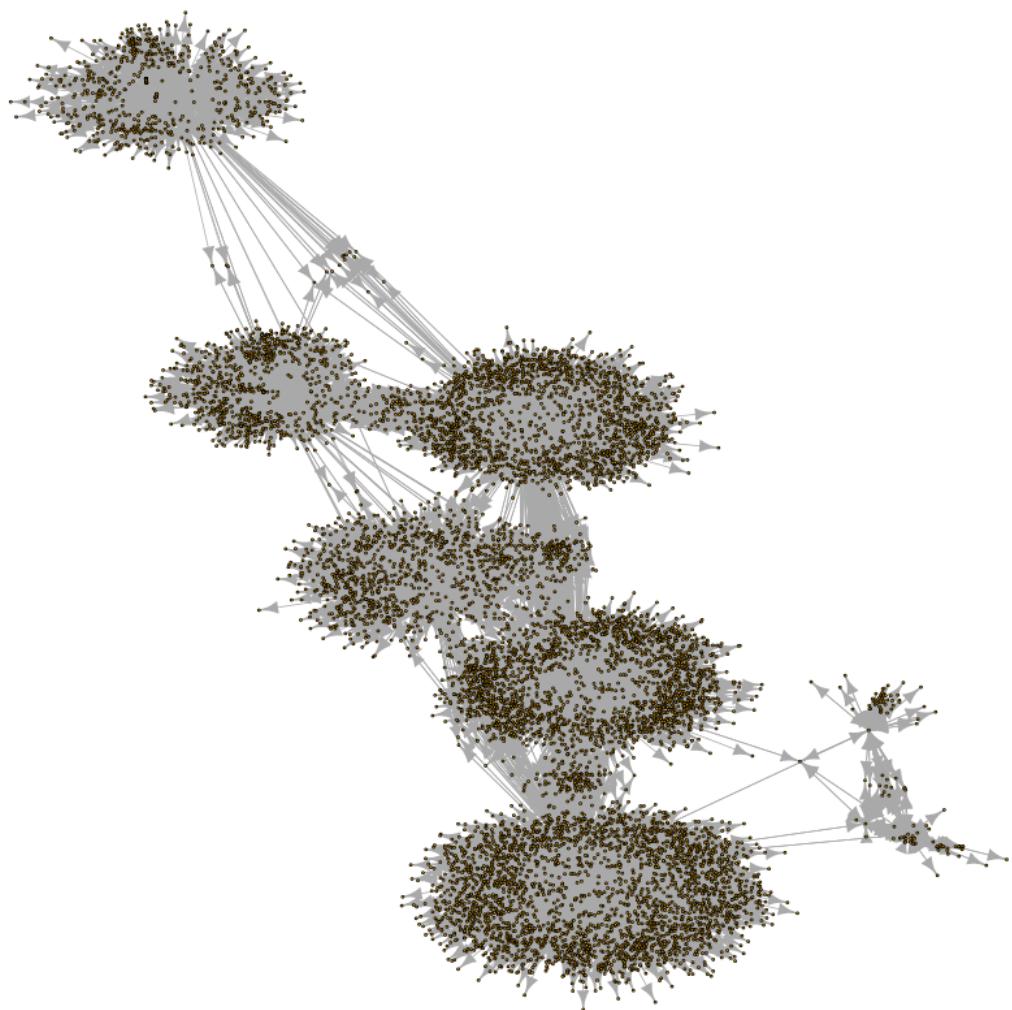


Figure 8: A sample connection graph started from one tweet.

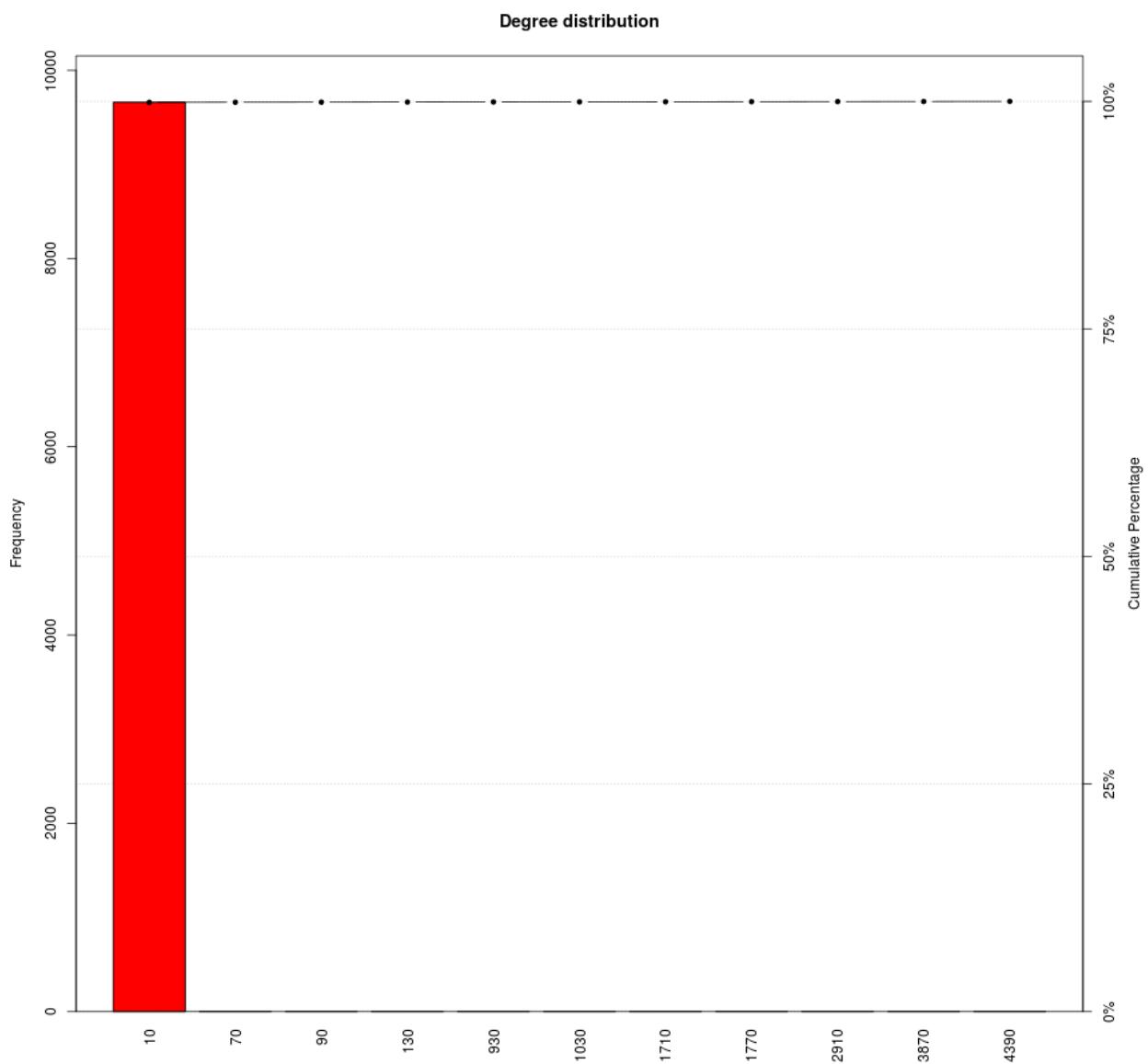


Figure 9: A Pareto graph of connections from the sample graph.

**167145 unique nodes, 273431 edges**

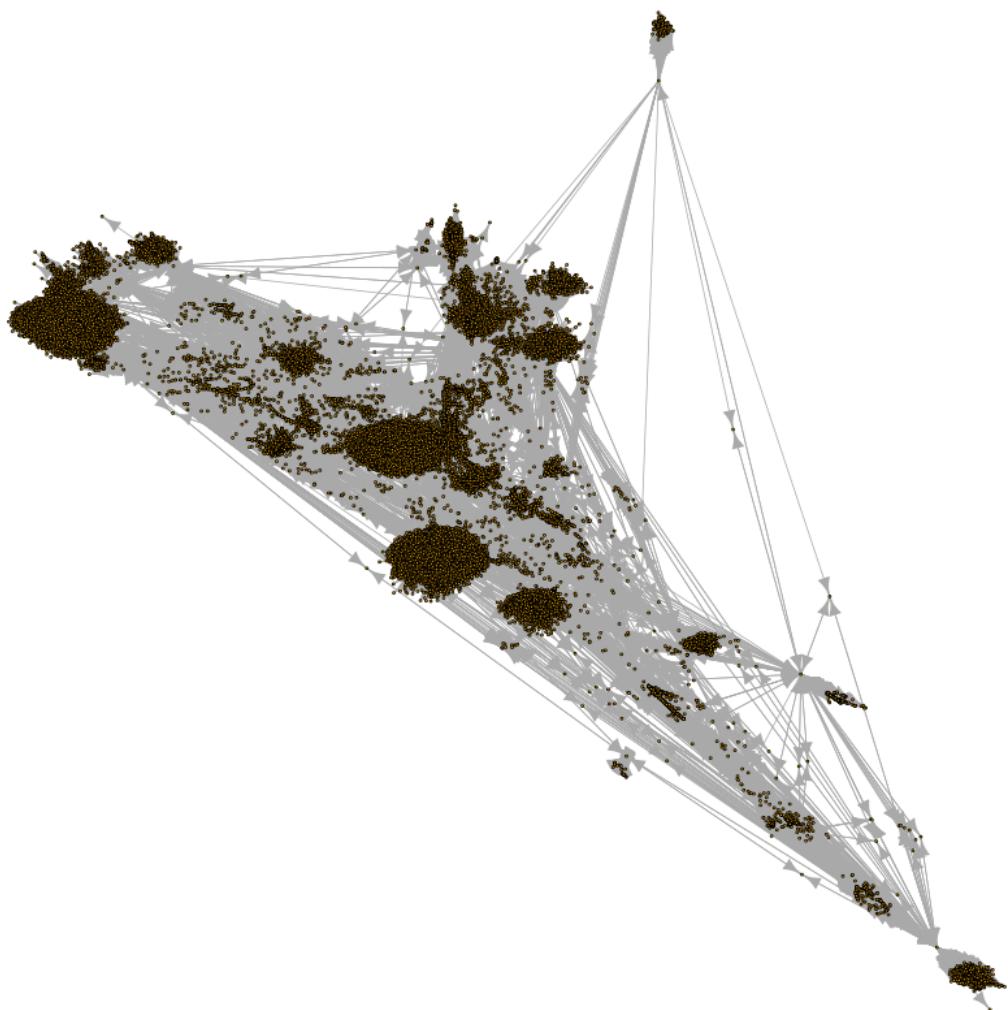


Figure 10: A connection graph including all discovered tweeters.

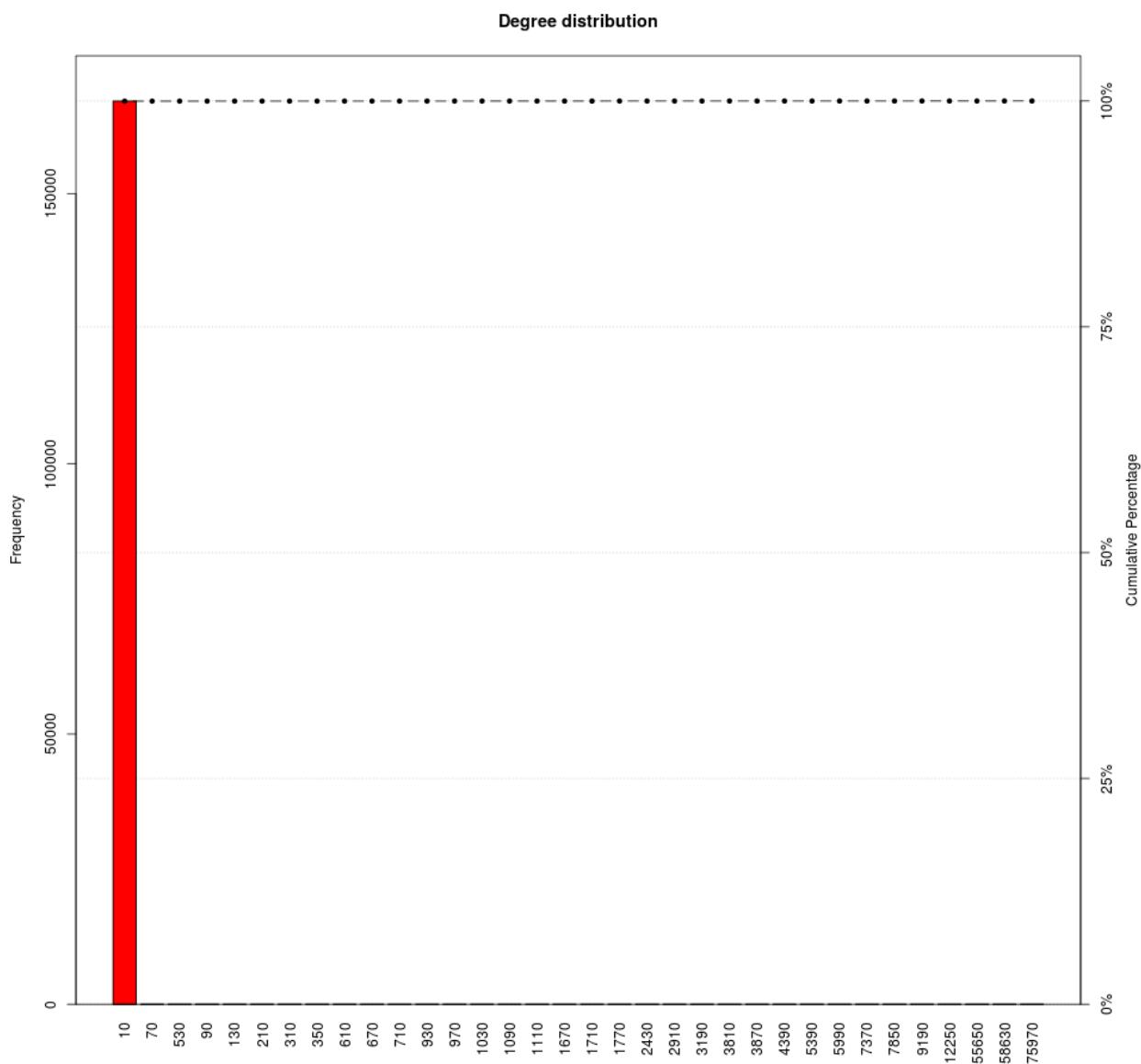


Figure 11: A Pareto graph of connections including all discovered tweeters.

Table 1: Most well connected tweeter of sample tweets. *The entries are ordered by the number of edges that are incident to each node. Only 30 most connected nodes are shown.*

#	Edges	Name	Description	Location
1	4,393	001SPARTaN	Hacker, speedskater, DJ, feminist, gamer. Hacking smarter, skating faster. Always learning. I work somewhere. Tweets are mine, don't blame my employer.	root@127.0.0.1: #
2	3,880	0x1C	vi evangelist, former cyber thought leader, ve/vim/vis	:!pwd
3	2,918	00Syssiphus	Coder	Peine, Germany
4	1,766	00k	Authorised by the secretary of the 'Vote For @00k' party	Christchurch, New Zealand
5	1,707	01acsap	lang not English, data ignored	lang not English, data ignored
6	1,031	0b10111	Data Science and InfoSec	
7	921	anitayorker	PhD student at UCLA, interested in Data Mining and machine translation	Los Angeles, California
8	138	0793211572		
9	96	0322199939		obuasi
10	74	04Aprilia		
11	12	Snowden	I used to work for the government. Now I work for the public. Director at @FreedomofPress.	
12	10	iamdevloper	These views are also the opinions of my employers, any problems, speak to their legal team. *puts headphones back in*	Localhost

(Continued on the next page.)

Table 1. (Continued from the previous page.)

#	Edges	Name	Description	Location
13	10	moxie		San Francisco, CA
14	10	thengrugq	Security Researcher :: Cultural Attach :: PGP https://t.co/JAjLPrVq1q :: , ,	http://grugq.github.io
15	8	elonmusk	Tesla, SpaceX, SolarCity & PayPal	1 AU
16	8	hmason	Founder at @FastForwardLabs. Data Scientist in Residence at @accel. I ♡ data and cheeseburgers.	NYC
17	8	letsencrypt	A free, automated, and open Certificate Authority (CA).	San Francisco, CA
18	8	neiltyson	Astrophysicist	New York City
19	8	runasand	Director of Information Security, Newsroom at @nytimes.	New York, NY
20	8	SpaceX	Official Twitter for SpaceX, the future of space travel. SpaceX designs, manufactures and launches the worlds most advanced rockets and spacecraft.	Hawthorne, CA
21	8	twitter	Your official source for news, updates and tips from Twitter, Inc. Need help? Visit http://t.co/qq1HEzvnra.	San Francisco, CA
22	8	whispersystems	Making private communication simple. For support, questions, or more information, please visit https://t.co/ALuyayfJfY	
23	7	kyhwana	Spottycat, atheist, discordian, geek, beer snob, queer, poly, hacker, weirdo, misanthrope, iconoclast, infosec, furry.	Auckland, NZ
24	6	0xcharlie	I'm that Apple 0day guy	St. Louis, MO
25	6	41414141		
26	6	agl__		

(Continued on the next page.)

Table 1. (Continued from the previous page.)

<b>#</b>	<b>Edges</b>	<b>Name</b>	<b>Description</b>	<b>Location</b>
27	6	antic0de	Network Intrusion Specialist FOLLOWS YOU. engineer at @brave.	
28	6	bcrypt	helped build @httpseverywhere @letsencrypt @securedrop @privacybadger @torproject. high school dropout / @MIT physics alum.	the internet
29	6	benhawkes	Project Zero	
30	6	brokep	(co-/)founder of @Flattr, @IpredatorVPN, @TPBdotORG, @KonstHack etc. Socialist, vegetarian, finnish	norwegian. French call me Le troll rveur. & Internets

(Last page.)

## 4 Conclusion

We have used real-time tweets as a surrogate for Doug Laney's velocity. We discussed how to get access to the tweets via a Twitter API, and once we have tweets, then we explored a few different ways to examine the sample data, and how to present some of the data we discovered.

We looked at the real-time tweets, and applied a sentiment analysis algorithm to determine if a tweet was positive, negative, or neutral. Based on those sample tweets, we were able to construct a graph showing which of the tweeters was very well connected compared to others. We then added that new data to an already existing graph to see how the twitterverse in general looked. In both cases, there are a very few well connected tweeters, and a vast collection who are not.

We took a look at some of the information available from these well connected tweeters and saw that the information they were willing to share with the twitterverse may not be really indicative of who or where they are. It is almost as if they wanted to be heard, but wanted to remain anonymous.

Tweets are fun to play with, easy to hold, if you break one, we consider it OK.

## A Anatomy of a Tweet

Tweets are the atomic building blocks of all things Twitter. Tweets, also known more generically as “status updates.” (This information drawn from [11].) Tweets are made up of nested data structures, parts of which (and in some cases) may be *NULL*. Tweets can be returned from the Twitter application program interface (API) as either extensible markup language (XML) documents, or JavaScript Object Notation (JSON) objects. The description of the various tweet data fields is implementation agnostic (see Table 2). It is also possible, and likely that additional data fields will be added as time goes on. Some anatomies of other twitter data fields are also listed:

- A user (see Table 3 on page 23),
- An entity (see Table 4 on page 27), and
- A place (see Table 5 on page 27).

The following tables are not all inclusive of the tweet data fields. They represent the necessary fields to support Big Data velocity exploration.

Table 2: Anatomy of a tweet.

Field	Type	Description
contributors	Collection of Contributors	Nullable. An collection of brief user objects (usually only one) indicating users who contributed to the authorship of the tweet, on behalf of the official tweet author.
coordinates	Coordinates	Nullable. Represents the geographic location of this Tweet as reported by the user or client application. The inner coordinates array is formatted as geoJSON (longitude first, then latitude).
created_at	String	UTC time when this Tweet was created.
current_user_retweet	Object	Perspectival. Only surfaces on methods supporting the include_my_retweet parameter, when set to true. Details the Tweet ID of the users own retweet (if existent) of this Tweet.
entities	Entities	Entities which have been parsed out of the text of the Tweet (see Table 4 on page 27).

(Continued on the next page.)

Table 2. (Continued from the previous page.)

Field	Type	Description
favorite_count	Integer	Nullable. Indicates approximately how many times this Tweet has been liked by Twitter users.
favorited	Boolean	Nullable. Perspectival. Indicates whether this Tweet has been liked by the authenticating user.
filter_level	String	Indicates the maximum value of the filter_level parameter which may be used and still stream this Tweet.
id	Int64	The integer representation of the unique identifier for this Tweet. This number is greater than 53 bits and some programming languages may have difficulty/silent defects in interpreting it. Using a signed 64 bit integer for storing this identifier is safe. Use id_str for fetching the identifier to stay on the safe side.
id_str	String	The string representation of the unique identifier for this Tweet. Implementations should use this rather than the large integer in id.
in_reply_to_screen_name	String	Nullable. If the represented Tweet is a reply, this field will contain the screen name of the original Tweets author.
in_reply_to_status_id	Int64	Nullable. If the represented Tweet is a reply, this field will contain the integer representation of the original Tweets ID.
in_reply_to_status_id_str	String	Nullable. If the represented Tweet is a reply, this field will contain the string representation of the original Tweets ID.
in_reply_to_user_id	Int64	Nullable. If the represented Tweet is a reply, this field will contain the integer representation of the original Tweets author ID.
in_reply_to_user_id_str	String	Nullable. If the represented Tweet is a reply, this field will contain the string representation of the original Tweets author ID.

(Continued on the next page.)

Table 2. (Continued from the previous page.)

Field	Type	Description
lang	String	Nullable. When present, indicates a BCP 47 language identifier corresponding to the machine-detected language of the Tweet text, or “und” if no language could be detected.
place	Places	Nullable. When present, indicates that the tweet is associated (but not necessarily originating from) a Place (see Table 5 on page 27).
possibly_sensitive	Boolean	Nullable. This field only surfaces when a tweet contains a link. The meaning of the field doesn't pertain to the tweet content itself, but instead it is an indicator that the URL contained in the tweet may contain content or media identified as sensitive content.
quoted_status_id	Int64	This field only surfaces when the Tweet is a quote Tweet. This field contains the integer value Tweet ID of the quoted Tweet.
quoted_status_id_str	String	This field only surfaces when the Tweet is a quote Tweet. This is the string representation Tweet ID of the quoted Tweet.
quoted_status	Tweet	This field only surfaces when the Tweet is a quote Tweet. This attribute contains the Tweet object of the original Tweet that was quoted.
scopes	Object	A set of key-value pairs indicating the intended contextual delivery of the containing Tweet.
retweet_count	Int	Number of times this Tweet has been retweeted. This field is no longer capped at 99 and will not turn into a String for “100+”
retweeted	Boolean	Perspectival. Indicates whether this Tweet has been retweeted by the authenticating user.
retweeted_status	Tweet	Users can amplify the broadcast of tweets authored by other users by retweeting.
source	String	Utility used to post the Tweet, as an HTML-formatted string. Tweets from the Twitter website have a source value of web.

(Continued on the next page.)

Table 2. (Continued from the previous page.)

Field	Type	Description
text	String	The actual UTF-8 text of the status update.
truncated	Boolean	Indicates whether the value of the text parameter was truncated, for example, as a result of a retweet exceeding the 140 character Tweet length.
user	Users	The user who posted this Tweet (see Table 3). Perspectival attributes embedded within this object are unreliable.
withheld_copyright	Boolean	When present and set to true, it indicates that this piece of content has been withheld due to a DMCA complaint.
withheld_in_countries	Array of String	When present, indicates a list of uppercase two-letter country codes this content is withheld from.
withheld_scope	String	When present, indicates whether the content being withheld is the “status” or a “user.”

(Last page.)

Table 3: Anatomy of a tweet:user.

Field	Type	Description
contributors_enabled	Boolean	Indicates that the user has an account with contributor mode enabled, allowing for Tweets issued by the user to be co-authored by another account.
created_at	String	The UTC datetime that the user account was created on Twitter.
default_profile	Boolean	When true, indicates that the user has not altered the theme or background of their user profile.
default_profile_image	Boolean	When true, indicates that the user has not uploaded their own avatar and a default egg avatar is used instead.

(Continued on the next page.)

Table 3. (Continued from the previous page.)

Field	Type	Description
description	String	Nullable. The user-defined UTF-8 string describing their account.
entities	Entities	Entities which have been parsed out of the url or description fields defined by the user (see Table 4 on page 27).
favourites_count	Int	The number of tweets this user has favorited in the accounts lifetime.
follow_request_sent	Type	Nullable. Perspectival. When true, indicates that the authenticating user has issued a follow request to this protected user account.
following	Type	Nullable. Perspectival. Deprecated. When true, indicates that the authenticating user is following this user.
followers_count	Int	The number of followers this account currently has. Under certain conditions of duress, this field will temporarily indicate 0.
friends_count	Int	The number of users this account is following (AKA their followings). Under certain conditions of duress, this field will temporarily indicate 0.
geo_enabled	Boolean	When true, indicates that the user has enabled the possibility of geotagging their Tweets.
id	Int64	The integer representation of the unique identifier for this User. This number is greater than 53 bits and some programming languages may have difficulty/silent defects in interpreting it. Using a signed 64 bit integer for storing this identifier is safe.
id_str	String	The string representation of the unique identifier for this User. Implementations should use this rather than the large, possibly un-consumable integer in id.
is_translator	Boolean	When true, indicates that the user is a participant in Twitters translator community.

(Continued on the next page.)

Table 3. (Continued from the previous page.)

Field	Type	Description
lang	String	The BCP 47 code for the users self-declared user interface language. May or may not have anything to do with the content of their Tweets.
listed_count	Int	The number of public lists that this user is a member of.
location	String	Nullable. The user-defined location for this accounts profile. Not necessarily a location nor parseable. This field will occasionally be fuzzily interpreted by the Search service.
name	String	The name of the user, as theyve defined it. Not necessarily a persons name.
notifications	Boolean	Nullable. Deprecated. May incorrectly report false at times. Indicates whether the authenticated user has chosen to receive this users tweets by SMS.
profile_background_color	String	The hexadecimal color chosen by the user for their background.
profile_background_image_url	String	A HTTP-based URL pointing to the background image the user has uploaded for their profile.
profile_background_image_url_https	String	A HTTPS-based URL pointing to the background image the user has uploaded for their profile.
profile_background_tile	Boolean	When true, indicates that the users profile_background_image_url should be tiled when displayed.
profile_banner_url	String	The HTTPS-based URL pointing to the standard web representation of the users uploaded profile banner.
profile_image_url	String	A HTTP-based URL pointing to the users avatar image.
profile_image_url_https	String	A HTTPS-based URL pointing to the users avatar image.
profile_link_color	String	The hexadecimal color the user has chosen to display links with in their Twitter UI.

(Continued on the next page.)

Table 3. (Continued from the previous page.)

Field	Type	Description
profile_sidebar_border_color	String	The hexadecimal color the user has chosen to display sidebar borders with in their Twitter UI.
profile_sidebar_fill_color	String	The hexadecimal color the user has chosen to display sidebar backgrounds with in their Twitter UI.
profile_text_color	String	The hexadecimal color the user has chosen to display text with in their Twitter UI.
profile_use_background_image	Boolean	When true, indicates the user wants their uploaded background image to be used.
protected	Boolean	When true, indicates that this user has chosen to protect their Tweets.
screen_name	String	The screen name, handle, or alias that this user identifies themselves with. screen_names are unique but subject to change.
show_all_inline_media	Boolean	Indicates that the user would like to see media inline.
status	Tweets	Nullable. If possible, the users most recent tweet or retweet.
statuses_count	Int	The number of tweets (including retweets) issued by the user.
time_zone	String	Nullable. A string describing the Time Zone this user declares themselves within.
url	String	Nullable. A URL provided by the user in association with their profile.
utc_offset	Int	Nullable. The offset from GMT/UTC in seconds.
verified	Boolean	When true, indicates that the user has a verified account.
withheld_in_countries	Array of String	When present, indicates a list of uppercase two-letter country codes this content is withheld from.
withheld_scope	String	When present, indicates whether the content being withheld is the “status” or a “user.”

(Last page.)

Table 4: Anatomy of a tweet:entity.

<b>Field</b>	<b>Type</b>	<b>Description</b>
hashtags	Array	of Object Represents hashtags which have been parsed out of the Tweet text.
media	Array	of Object Represents media elements uploaded with the Tweet.
urls	Array	of Object Represents URLs included in the text of a Tweet or within textual fields of a user object.
user_mentions	Array	of Object Represents other Twitter users mentioned in the text of the Tweet.

Table 5: Anatomy of a tweet:places.

<b>Field</b>	<b>Type</b>	<b>Description</b>
attributes	Object	Contains a hash of variant information about the place.
bounding_box	Object	A bounding box of coordinates which encloses this place.
country	String	Name of the country containing this place.
country_code	String	Shortened country code representing the country containing this place.
full_name	String	Full human-readable representation of the places name.
id	String	ID representing this place.
name	String	Short human-readable representation of the places name.
place_type	String	The type of location represented by this place.

(Continued on the next page.)

Table 5. (Continued from the previous page.)

<b>Field</b>	<b>Type</b>	<b>Description</b>
url	String	URL representing the location of additional place metadata for this place.

(Last page.)

Table 6: Real-time software and system classifications. *These classifications are more qualitative than quantitative, because their respective timeliness is dependent on the needs of the user.*

<b>Classification</b>	<b>Definition</b>	<b>Notional examples</b>
Hard real-time	The response time is specified as an absolute value. A system is called a hard real-time if tasks always must finish execution before their deadlines or if message always can be delivered within a specified time interval.	Engine control in modern cars.
Soft real-time	The response time is normally specified as an average value. A single computation arriving late is not significant to the operation of the system, though many late arrivals might be.	Downloading a web page.
Non real-time	The response time is “reasonable.”	A complicated Excel spreadsheet.

## B Real-time software

In general there are two main ways to evaluate the correctness of a computer program or system. They are:

1. Is the correct answer, or solution, or service provided, and
2. Is the answer, or solution, or service provided in time.

The first criterion can be confirmed by evaluating what is delivered by the system as compared with what is expected. While the second criterion can be more difficult to quantify because it has a temporal aspect. Programs or systems that have a temporal constraint, are generally called “real-time.” Real-time systems that meet the correctness criterion, but not the timeliness criterion are considered failures[7].

Real-time systems can be classified based on how tightly their correctness is tied to their timeliness (see Table 6)[5].

## C Misc. files

The files used to create all these figures are attached to this report. They are:

1. lexicon.csv - the list of positive and negative words 
2. connections.R - an R program used to discover and record connections (uses a postgres database) 
3. twitter03.R - an R program that implements the above algorithm and plots the results 
4. twitterLibrary.R - a support R file used by twitter03.R and connections.R 
5. psqlCommands.sql - the postgres commands to create the local twitter connection database 

The twitter key and secret strings that are associated with your application will need to be added to twitterLibrary.R in the keyAndSecret() function (around line 3) for your code to work.

## D References

- [1] Pablo Barberá, *Nyu politics data lab workshop: Scraping twitter and web data using r*, 2013.
- [2] Wei Fan and Albert Bifet, *Mining big data: Current status, and forecast to the future*, ACM SIGKDD Explorations Newsletter **14** (2013), no. 2, 1–5.
- [3] Doug Laney, *3d data management: Controlling data volume, velocity and variety*, META Group Research Note **6** (2001).
- [4] Doug Laney, *Deja vvvu: Others claiming gartners construct for big data*, <http://blogs.gartner.com/doug-laney/deja-vvvue-others-claiming-gartners-volume-velocity-variety-construct-for-big-data/>, 2012.
- [5] Insup Lee, *Cis 505: Software systems os overview real-time scheduling*, <http://www.cis.upenn.edu/~lee/07cis505/Lec/RT-scheduling-v2.pdf>, 2007.
- [6] Bo Pang and Lillian Lee, *Opinion mining and sentiment analysis*, Foundations and Trends in Information Retrieval **2** (2008), no. 1-2, 1–135.
- [7] Stefan M. Peters, *Real-time systems*, <http://www.cse.unsw.edu.au/~cs9242/08/lectures/09realtimex2.pdf>, 2009.
- [8] Anand Rajaraman, Jeffrey D Ullman, Jeffrey David Ullman, and Jeffrey David Ullman, *Mining of massive datasets*, vol. 1, Cambridge University Press Cambridge, 2012.
- [9] Bitly Staff, *Shorten. share. measure.*, 2016.
- [10] Twitter Staff, *New user faqs*, <https://support.twitter.com/articles/13920>, 2016.
- [11] ———, *Tweets, field guide*, <https://dev.twitter.com/overview/api/tweets>, 2016.
- [12] Wikipedia Staff, *Stop words*, [https://en.wikipedia.org/wiki/Stop\\_words](https://en.wikipedia.org/wiki/Stop_words), 2016.
- [13] ———, *Twitter*, <https://en.wikipedia.org/wiki/Twitter>, 2016.
- [14] Biz Stone, *Friends, followers, and notifications*, <https://blog.twitter.com/2007/friends-followers-and-notifications>, 2007.
- [15] Rob Triggs, *What is sms and how does it work?*, <http://www.androidauthority.com/what-is-sms-280988/>, 2013.