Stefania Maiman, Matt Schulman, Hannah Cutler, Calvin Casalino
NETS 150 Homework 5 - Final Project Analysis
May 5, 2014

Project Summary

Our project was a combination of implementation and empirical analysis, as we both wrote a program using Twitter's API and analyzed data gathered from the program and did analysis using concepts discussed in class (i.e. tf-idf analysis).

Overview:

Our program uses Twitter data to tell a user the "word on the street" for a given topic or series of arguments by generating a word cloud. Why a word cloud? A word cloud offers a visual presentation of text data, making it easy to spot text frequency in a document, as words with higher frequency are larger and bolder in the cloud. Word clouds allow a reader to easily identify trends that might be harder to detect or go unnoticed in a long, written document.

One of the drawbacks commonly associated with word clouds is that they analyze word frequency but aren't necessarily good indicators of importance or context of a word or set of words. So, many people argue that they are not as effective for exploring high-level topics such as (but not limited to) geopolitical issues or day-to-day fluctuations in the stock market. Using the raw data from Twitter (i.e. tweets, which are capped at 140 characters each) we are gathering the 'talk of the town' or, as mentioned above, the 'word on the street' about a particular topic, so presumably our word cloud can be used to bring a user up to speed on any topic they're curious about, in such a manner that is both user-friendly and quick. Unlike simply looking at a trending topic by navigating through all of the tweets containing a particular hashtag, we aimed to create a simple program able to navigate and explore a certain topic, enhancing a user's experience.

How it works:

1. Program asks user for 1 or more arguments
2. Program uses twitter's API to get the most recent tweets (up to 3,200) for each of the users' arguments
3. For each argument, a document is created with all of the text of the 3,200 tweets
4. For each document, a hash map is created mapping each word to its frequency. The hash map will not include common words (i.e. "the", "a", "and", etc.).[1]
5. For each document, a word cloud will be generated and printed out for the user.
6. If the user inputs more than one argument, statistical analysis will be conducted using tf-idf and summary statistics about word frequency, etc.

Analysis:

Our analysis focused primarily on tf-idf/cosine similarity analysis[2], based on the following formulas:

[1] We are aware that tf-idf accounts for common words such as "the", "a", "and", etc. that are likely to appear frequently in both/all documents we test, but our program splits these words so that they are not included in the actual word cloud.

[2] Our program automatically computes the tf-idf for the raw Twitter data text files of the various arguments, using (and slightly modifying) the 'ir' code from Homework 4.

$$tf\text{-}idf_{term,\ document} = tf_{term,\ document} * idf_{term} \quad \textbf{and} \quad idf_{term} = log\left(\frac{N}{df\ term}\right)$$

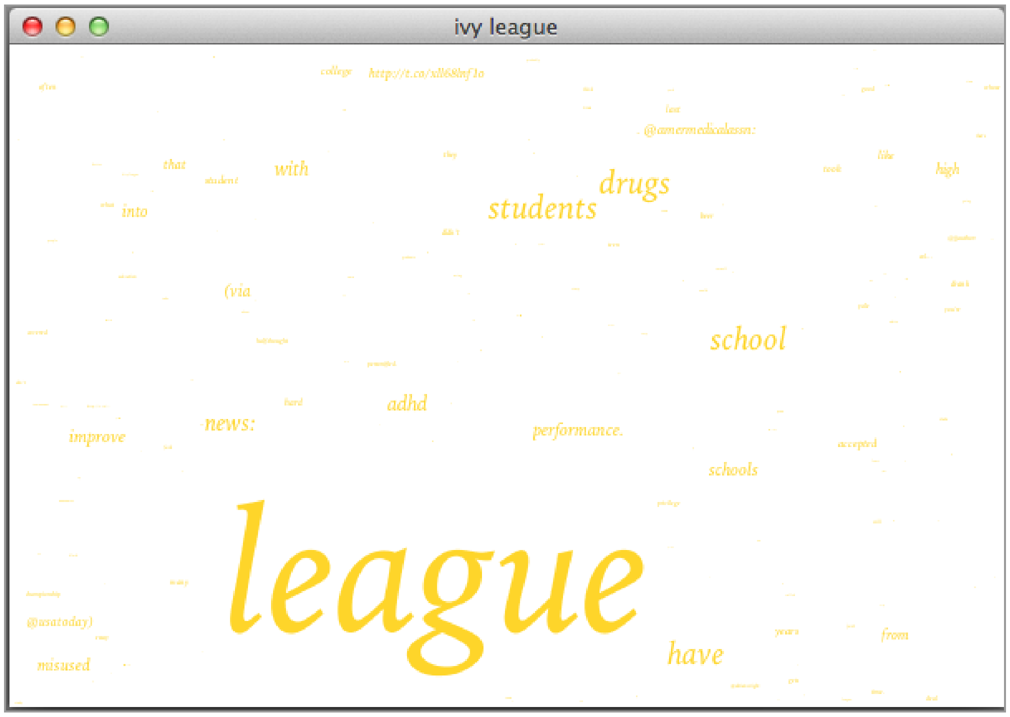
We were trying to gage the accuracy of our program, and of the word clouds in generating the "word on street", to inform people about a particular argument using information from Twitter. If a user is curious about the Kentucky Derby, the results of the most recent sports game between, say, Princeton and Penn, the latest news about Ukraine, will our program (i.e. the word cloud) theoretically do an adequate job of informing the user so that they don't really need to bother reading a newspaper? As a follow up, what might the results of our analysis say about Twitter as a source of information, and the nature of how information spreads in such a modern-day social network?

- We ran tf-idf analysis on the raw Twitter data text files for all eight Ivy League schools. Our results were the following: [3]

	Brown	Columbia	Dartmouth	Harvard	Yale	UPenn	Cornell	Princeton
Brown		0.006507492247508764	0.011871159847090886	0.01448970960253617	0.012221807561524675	0.006479175409158981	0.0017270974167033165	0.003382863008353275
Columbia			0.006807921165512696	0.006856157331068763	0.009849707771882174	0.003174171253115924	0.007591130036785288	7.629400641827374E-4
Dartmouth				0.010533474323598139	0.02164085379703808	0.0017638322104134856	1.2062745547227808E-4	0.0018923616413511319
Harvard					0.055326890346962546	0.0031885020465425833	0.0024291481364666005	0.041053287163478086
Yale						0.008851534105993987	0.005062792663976451	0.01725195456030285
UPenn							1.038618651026878E-5	1.8210917036812777E-4
Cornell								1.1005095194260827E-4
Princeton								

[3] **Yellow** *indicates similarity > 0.02,* **Grey** *indicates similarity > 0.01*

Below is the **word cloud** our program generated for the arguments “ivy league” and “smart”:

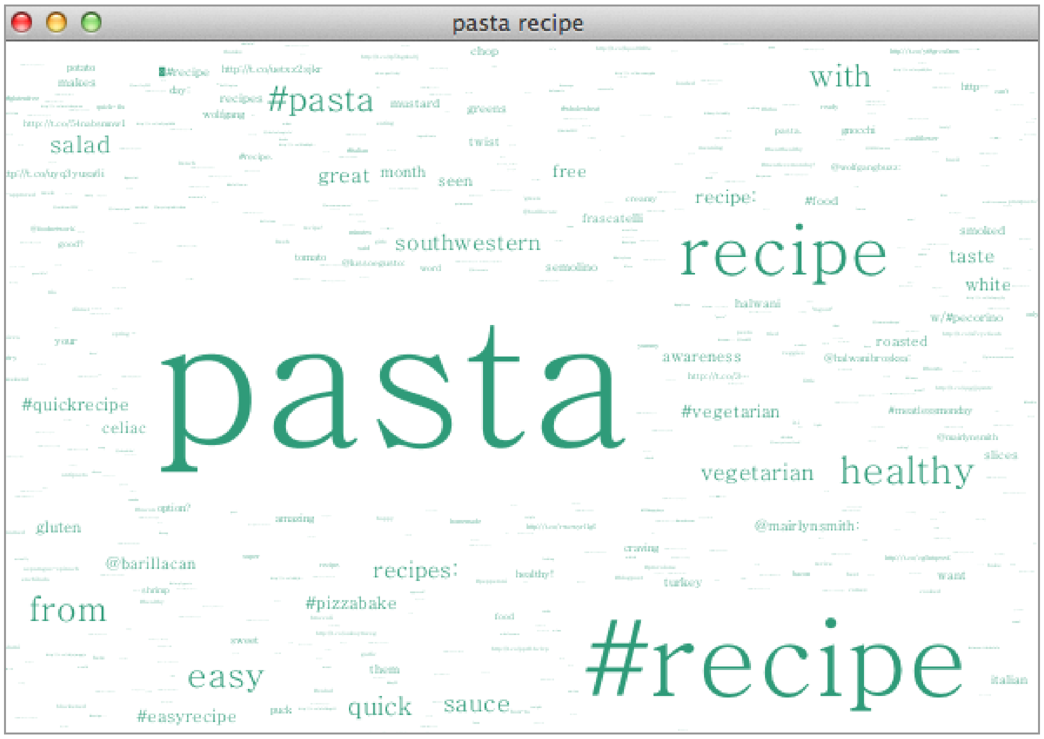


- Second, we ran tf-idf analysis to compare the raw Twitter data for “Philadelphia” “foodie” and “New York” against the query “pasta recipe”.[4][5]
  *Questions: *Are there more foodies in Philadelphia than in New York? Can we gain any information about a pasta recipe, or pasta in general from a word cloud, or is this a case where it’d be worth consulting a more reliable source like Bon Appetit?*

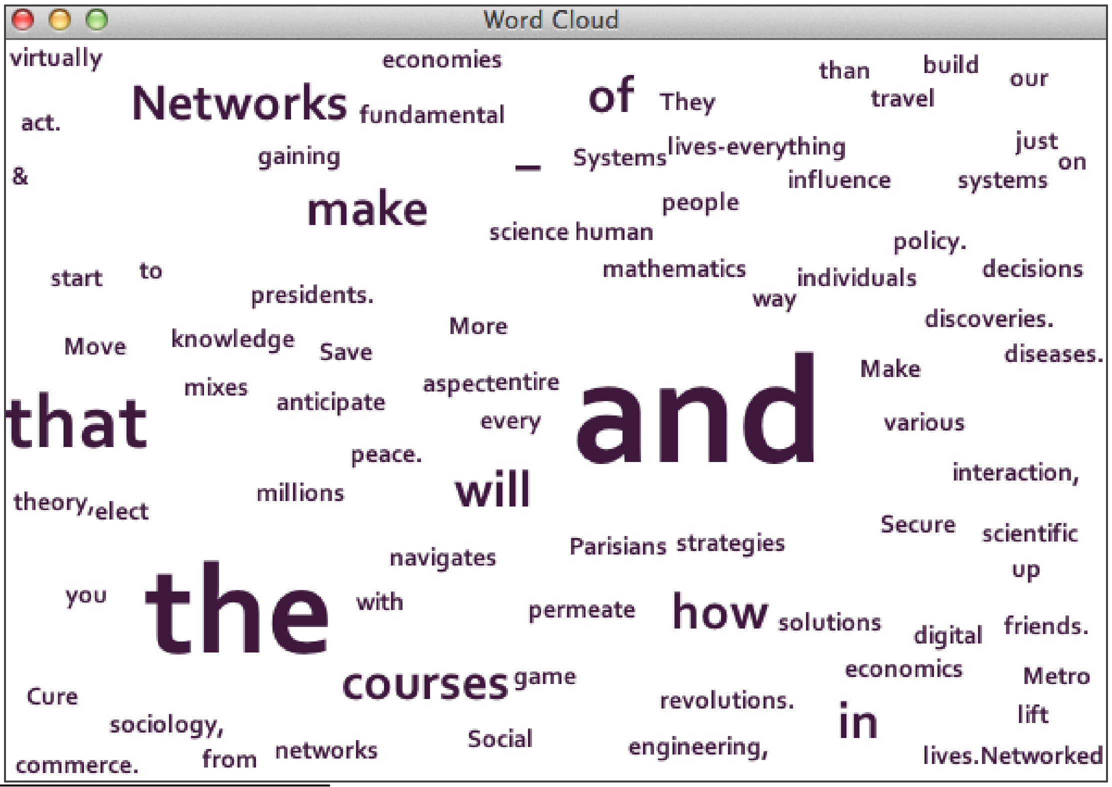
	Philadelphia	NewYork	foodie	‘query’
Philadelphia		0.023913369527759614	0.0330053178196 5382	0.0
NewYork			0.0186690269309 8034	0.0
foodie				0.051542588152864485
‘query’				

[4] inserted into query.txt file, which was added to the Corpus
[5] **Blue** *indicates similarity > 0.02,* **Grey** *indicates similarity > 0.01*

This is the **word cloud** our program generated for the arguments: “pasta recipe” and “healthy”:



Third, we generated the **word cloud** using the NETS major description listed on the NETS homepage:[6]



[6] Note: This NETS major word cloud was generated *before* we deleted the insignificant words such as “the”, “and”, “with”, etc. but general idea is still clear.

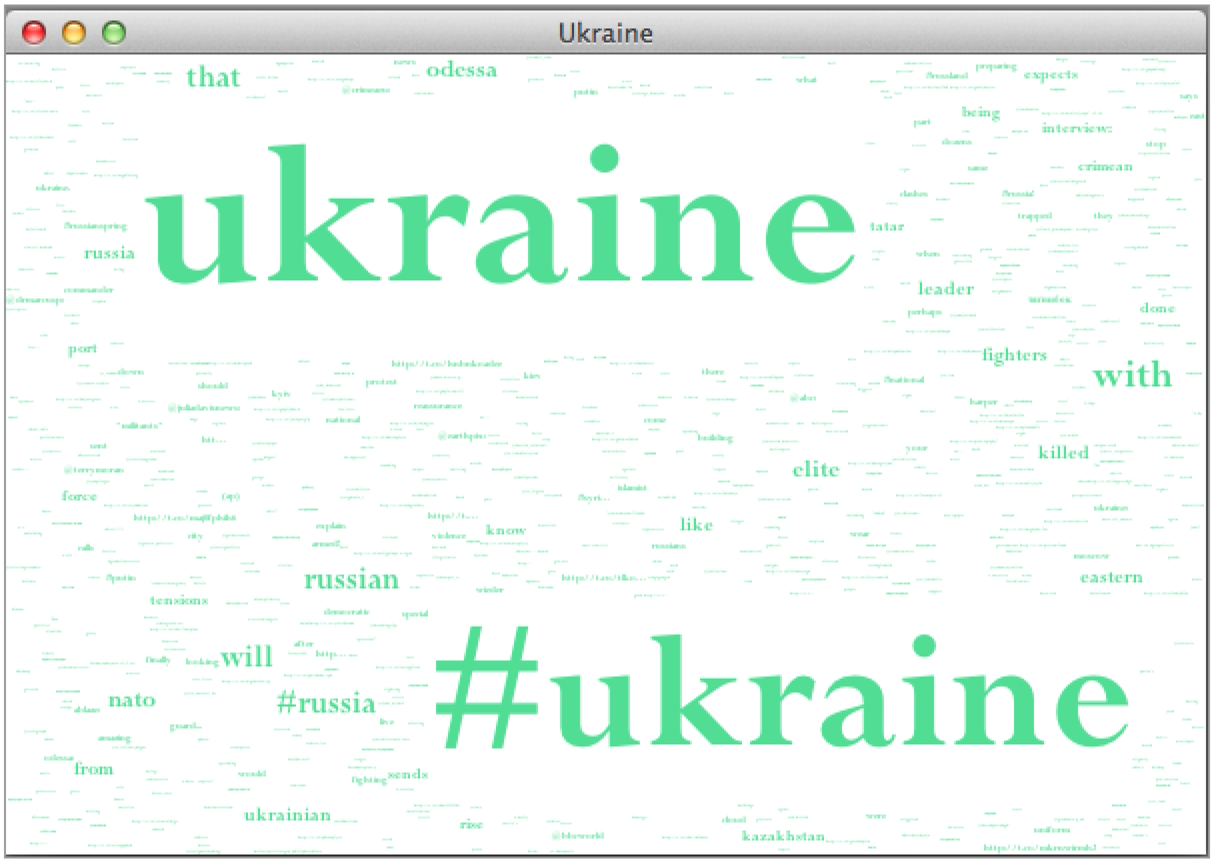
- Lastly, we tested the raw twitter data of the query "Ukraine" on May 4th and subsequently on May 5th against the top three news articles on Ukraine pulled on May 5th (from *The New York Times, Wall Street Journal, CNN,* and *Fox News*):[7]

	Ukraine (5/4)	Ukraine (5/5)	FoxNews_Ukraine	WSJ_Ukraine	NYTimes_Ukraine	CNN_Ukraine
Ukraine (5/4)		0.1798895521741642	0.01826716847283473	0.016109100026708192	0.010638766467939516	0.023955902004962212
Ukraine (5/5)			0.030287651407475635	0.02957533933208777	0.01623698409954952	0.036663068450318886
FoxNews_Ukraine				0.21461578664858696	0.1448272966556952	0.19465673333033104
WSJ_Ukraine					0.15370274743891862	0.17133775507910287
NYTimes_Ukraine						0.14194978647416795
CNN_Ukraine						

As the above table shows, the news articles have highest similarity to one another, and the *lower* similarity values correspond to the pairwise comparisons that include the file with Ukraine raw Twitter data. Initially, the Twitter data being used in the comparison tests was from May 4th, while the articles were all pulled on May 5th which may explain why the similarity was lower -- the Twitter data was already old news. To try and get more accurate results, we re-fetched the Twitter data for Ukraine on May 5th and tested this new data against all the articles and against the old data from 5/4. As expected, the newer Twitter data showed higher similarity to all the news articles, and interestingly, also showed high similarity, still, to the Twitter data from the prior day. The results of this test show that for our model to be most accurate, the comparisons should be run and the data should be pulled altogether, on the same day at the same time.

[7] **Red** *indicates >0.2*, **orange** *indicates 0.2 > cosine similarity > 0.1*, **green** *indicates 0.1 > cosine similarity > 0.02*

Below is the **word cloud** that our program generated for the arguments “Ukraine” and “news”:



Further analysis:

The results of our data hint that our word cloud generator may not inform a user about a particular topic as comprehensively as a full news article might, which we expected all along. Our pool of data is confined to the crowdsourced information that people are tweeting about a particular argument(s) at the moment the program is run. Overall however, it does do an adequate job at giving a user the “word on the street” about whatever they’re curious about. In a test trial on Sunday 5/4 around 1:30pm, we passed in the arguments “Princeton and “University of Pennsylvania”, and one of the words that appeared in the word cloud was “softball”. Little did we know (but thanks to the word cloud), Penn had played Princeton in softball on Saturday 5/3.

There are several factors that will influence the accuracy of this program, and which may have positively and/or negatively skewed our data. For example, it is important to consider the time of day our program is being used -- since we are collecting information from Twitter, we must recognize that people are most likely tweeting more frequently during the day than in the middle of the night, and this is relative to whatever region or time zone our arguments/queries are most relevant to. In other words, if we are searching “Paul Ryan” or “Boston Marathon”, people *in the United States* are likely tweeting more prolifically during hours of the day than people in say, Australia, so during this time frame the program is likely to be even more accurate. Another factor to consider is language. Our program doesn’t do any word translation, so if we search an argument about which people happen to be tweeting in a language besides English, then that particular word cloud will simply include non-English words because a word’s appearance or lack thereof in the word cloud is based on the term frequencies, which we store in a Hashmap when the program writes the raw Twitter data to an external text file on the user's computer. That said however, our program *does* include hashtags (#). Lastly, the tests included in our analysis were intended to be somewhat diverse to test the accuracy of our program on any kind of argument/query. With more extensive analysis, we might be able to confirm whether our program is perhaps more accurate on a particular type of query (if on average, more people tweet about it, there's a higher probability that there will be more relevant tweets and thus more data at the time that the program is run and the argument(s) is inputted).

Work Breakdown:
Matt/Calvin: Wrote all code for program (and modified 'WordCloud' code and Swap's tf-idf code)
Steffi/Hannah: Data analysis using program and write-up.