Geobio Boo

Twitter Assignment 2

NOTE: I did all of the work in Java, not in Pig. Java was the logical language for most of the work, but as I progressed I found a few instances where Pig would have been much simpler for manipulating data (especially with Joins and Chi Squared, and finding most popular words). However, I just finished the work in Java.

NOTE 2: For each question, I reference the source code file that was used at the end of the answer. All source code has Javadoc that explains what it does. In each source code, I manually set the file names and a few paramters (eg the limit of things to print, or the threshhold before a word is considered), so if you're running it, you can choose to change a few paramters. Files are read from and written to the top level project directory.

Question 1:

a) 54592 tweets over 20 mintes (45 tweets/sec), taken at 12pm Friday Sept 28. However some of them were empty or in foreign languages. Empty tweets (per my online search) seem to be caused by some encoding or unsupported characters. Tweets with foreign language characters (eg chinese) show up as ????, so I filtered out tweets with 3+ sequential ?'s, and empty tweets. The resulting filtered list is 44578 tweets (reduction of 18%)

Source: PrintSampleStream.java & FilterTweets.java.

Output: q1-data-friday-12pm/twitterStream.txt & twitterStreamFiltered.txt

b) Using the same data from part A (aka 20 mins, friday noon), the results were:

the = 5518

of = 2087

lol = 894

omg = 219

berkeley = 0

stanford = 0

twitter = 311

facebook = 117

happy = 270

sad = 56

obama = 47

romney = 24

Source: WordCount.java

Output: printed to System.out via printWordsFor1b() in WordCount.java

Note: some processing was done on the words: Words are defined as alphanumeric (so characters with accents are excluded/split), and URL's were removed, and so was 'RT'. However, hashtags and usernames are kept with the @ or #.

c) Using the same data from part A, the top 10 terms are as follows:

minimum length = 2:

the=5518

to=5113

you=4805

de=4430

me=3965

que=3626

and=2899

in=2790

my=2752

no=2691

minimum length = 4:

that=1803

this=1385

just=1315

your=1298

with=1286

like=1075

have=1031

love=944

what=838

when=796

minimum length = 5:

about=560

today=521

follow=497

people=436

there=387

think=354

going=340

tonight=338

@justinbieber=329

really=325

Source: WordCount.java

Output: q1-data-friday-12pm/all-words-l2.txt & all-words-l5.txt (generated by changing MIN_WORD_LENGTH in WordCount.java)

d) Using the same time period as A, I did the test with words of minimum length 2, and minimum length 5.

I believe using a long time period (eg, longer than 20 mins) will reduce the number of words seen only once.

For min word length = 2:

91593 words, with 68174 of those appearing only once.

For min word length = 5:

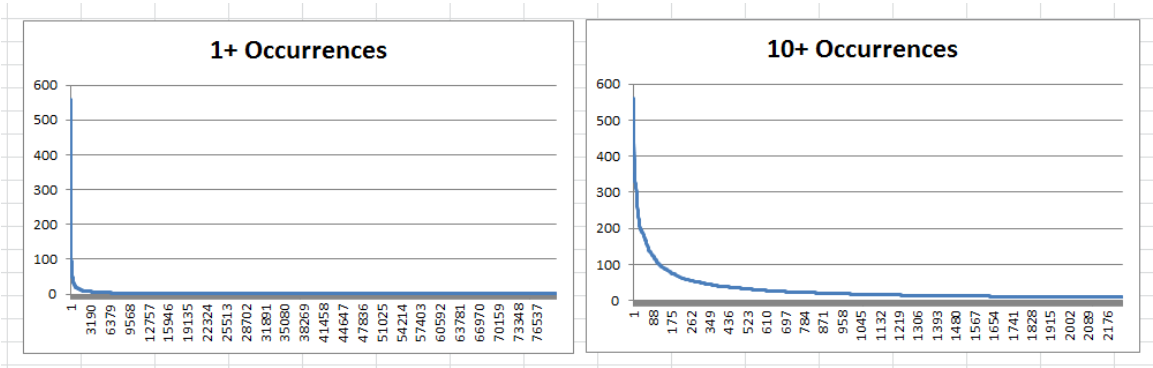
79713 words, with 62562 of those appearing only once

The list of words and their counts are in all-words-l2/5.txt. The background model is not outputted explicitly, but created when Chi Squared is calculated.

Plot: <See included picture chart for distribution>, or described in words, it's exponentially decreasing in occurrences.

Source: WordCount.java

Output: Same as in (c). Plotted in Excel and saved as a picture, 1d-plot.png



e) The new time period is Saturday 10pm, for 30 minutes. Because we Friday's collection had 20% less tweets, I scaled the count of words down by 20% for Saturday to compensate. This resulting list is the expected count for each word. From this, I selected words that showed up at least 100 times in both Friday and Saturday's collection (although it's easy to choose all words, but I don't trust the accuracy as much for such short time periods), and calculated the chi squared. The biggest chi squareds (and expected/observed values) are:

justin=1708, E=140 O=629

believe=266, E=106 O=274

night=216, E=280 O=526

hahaha=174, E=202 O=390

today=108, E=521 O=283

right=75, E=311 O=464

@justinbieber=63, E=329 O=184

doing=60, E=249 O=126

video=41, E=247 O=146

nigga=33, E=119 O=182

Source: ChiSquared.java (to run, have the required files in the top level directory)

Output: Printed to System.out

f) Comparing this to the list in C, I find that there's a lot more interesting words showing up, since it shows which words recently became more popular. Surprisingly, 'justin' showed up a lot, so I searched in the original tweets to find that Justin Bieber puked on stage at a concert, and that he was holding a concert for Avalanna Routh who passed away from cancer. The next word, 'believe' is the name of Justin Bieber's album and song. A search for 'night' didn't show any interesting events, so I attribute it to the time I collected tweets (10pm vs noon).

Question 2:

g) I wrote code in Following.java which pulls my followers, and then looks up their status updates (in groups of up to 100), and saves it in myfollowing.txt. Some updates were:

MJShores - MJ on the MBA is out! http://t.co/iMof3H09 ? Top stories today via @MBASCG @mba_exchange @cristianliu

ukrecruiter - @ChrisHeron1975 But I wonder if even something that out of reach from a £ perspective may inpsire other emps/rec's to try for the same?

marius - @niallohiggins mapped, or physical? Can you toggle its JVM switches?

MeiMeiFox - Esalen Institute turns 50 this year http://t.co/SDHl9uVs via @ChipConley. I love this magical place...

NovaNationBLS - Congratulations to @STVDayofService on a successful #STVC2012!

devironJ - I just got a $2 credit for music from @amazonmp3 and @imdb. Get your credit here: http://t.co/BUQPlQg0

mark_yudof - It is time to re-affirm an active, immutable commitment to academic quality at UC. Read my #UCRegents remarks http://t.co/0oTAgezQ

Source: Following.java

Output: q2-data/myfollowing.txt

h) I chose Fred Wilson (@fredwilson) and Michael Bloomberg. Fred Wilson is a VC in New York who writes a popular tech/venture-capitalist blog at http://avc.com. Michael Bloomberg is the mayor of New York. Fred Wilson has 233,000 followers, and Michael Bloomberg has 309,000 followers. I only pulled 100k followers from each of them, although with a argument change in the code, I can pull the full list (I reduced the count to avoid the Twitter API throttling). Of the 100k users, there were 3720 shared users. I checked 100 of those users (although it's easy to check on more/all), and saved some of their info into info_about_intersection.txt. As expected, there was slightly higher representation (18/100 of users from New York (since Fred Wilson and Michael Bloomberg are based in New York).

Source: UserIntersection.java & InfoAboutIntersection.java

Output: fredfollowers.txt, bloombergfollowers.txt, intersection.txt (the users that follow fred and bloomberg), & info_about_followers.txt (contains screen_name, name, location, latest-tweet)