CHICAGO BOOTH

BUSN 42118-81
Business Applications of Natural Language Processing

Spring 2018
Dean Alderucci

Final Project

Jack Gang

“I pledge my honor that I have not violated the Booth Honor Code in preparation of this assignment."

Table of Contents

Introduction

The general idea of this project is to perform geographic data visualization on sports and politics, topics that people tweet relatively often about, filtered by state and whether a user's name/username is likely to be male or female gendered.

One NLP technique used in this project is the topic modeling of the tweets to sports and politics topics. This was straightforward to do with regular expressions matching on a list of features. For example, if the tweet mentions certain politicians' names, it is likely about politics.

To identify the U.S. state of each user that only puts city name in his/her profile, I parsed out the location in each user's profile and performed a lookup in the geotargeting CSV mentioned above. If there is a collision (user profile says "Princeton" but there is a Princeton, NJ and a Princeton, WV) or if the city is not found, I discarded the user. Of course, users that indicate their state or state abbreviation in their profile were trivial cases.

For name gender, I parsed out the first names of each Twitter user and performed a lookup in a CSV of baby names that matches the name to the gender. If this was unsuccessful, I then used regular expressions to look for first names in their usernames (for example, username = LilyPichu could be a female name), since many twitter users do not reveal their real first name in their public profile. It's important to note that this NLP technique cannot actually say whether a user is male or female, just whether their name is typically a male or female name. In other words, this project will not be able to indicate what gender people actually identify as, only their name. This is an important distinction and is why I will have all three categories in the analysis: male-gendered names, female-gendered names, and ambiguous.

In the end, this project includes the full completed tool to take a tweet and return the following information if available:

- State in the U.S.
- Name gender of the user
- Count of politics-related and sports-related words

With the resulting dataset of this tool, I then outputted various statistics along with several "maps" visualizing the percentage of tweets by each state given various filters.

Data

The primary data source I used in this project is the selection of nearly thirty thousand tweets I accessed through the Twitter Developer API as in Assignment 2. I specifically only analyzed tweets by users who have their location set on their profile, since not all users populate this field.

To identify which state each user belongs to, I used a geotargeting CSV (https://developers.google.com/adwords/api/docs/appendix/geotargeting?csw=1) at the Google Adwords API with a filter on the U.S. country code. This way, I was also able to map users who only put their city name in their Twitter profile (though not perfectly, since there are collisions for city names across states and countries).

To identify each user name as a male or female name, I looked for their name in the following CSV: https://github.com/hadley/data-baby-names/blob/master/baby-names.csv. For usernames/names that could be categorized, they formed a third category ("ambiguous") for comparative analysis.

In addition to the programs described below, I've also included the following data with this final project:

- TweetResults.csv – a list of analysis results in the following format (tab-delimited): tweet text, tweet location, tweet user name, tweet screenname, analyzed location, analyzed name gender, # politics words, # sports words
- NewGeotargets.csv – CSV containing key-value pairs mapping city to state
- baby-names.csv – CSV containing key-value pairs mapping name to name gender
- Politics.csv – a 100-line annotated (1=correct, 0=ambiguous, -1=incorrect) subset of tweets that were classified as politics-related
- Sports.csv – a 100-line annotated subset of tweets that were classified as sports-related
- NameGender.csv – a 100-line annotated subset of user names and screennames that were classified as non-ambiguous gender names
- State.csv – a 100-line annotated subset of user locations that were classified as non-ambiguous states
- NotPolitics.csv – a 100-line annotated subset of tweets that were not classified as politics-related
- NotSports.csv – a 100-line annotated subset of tweets that were not classified as sports-related
- NotState.csv – a 100-line annotated subset of user locations that were classified as ambiguous states

Results and Takeaways

Data collection

First, the initial three filters I used when collecting tweets were:

1. The user has to be non-null
2. The user location has to have a value
3. The user's language has to be English

This filter resulted in discarding 35% of the data before any analysis was performed.

State

Out of the 28,037 tweets' user locations analyzed, this tool was able to identify a U.S. state 9,782 (35%) times. A manual annotation of 100 classifications found 2 false positives and 2 ambiguous locations, giving the model a precision of 0.96. Likewise, a manual annotation of 100 non-classifications found 2 false negatives, giving the model a recall of 0.98. Therefore, the F1 score for the state identification model is 0.97.

Name Gender

Out of the 28,037 tweets' user names and screennames analyzed, this tool was able to identify a name gender 9,378 (33%) times. A manual annotation of 100 classifications found 5 false positives and 12 ambiguous locations, giving the model a precision of 0.83. It is difficult to manually determine a recall for this model, as many names are surprisingly ambiguous (for example, there have been girls named "Michael" and boys named "Elizabeth") according to the historical baby name data.

Sports

Out of the 28,037 tweets analyzed, this tool was able to identify sports-related tweets 854 (3%) times. A manual annotation of 100 classifications found 10 false positives and 7 ambiguous locations, giving the model a precision of 0.83. Likewise, a manual annotation of 100 non-classifications found no false negatives, giving the model a recall of 1.00. Therefore, the F1 score for the sports topic identification model is 0.91.

Politics

Out of the 28,037 tweets analyzed, this tool was able to identify politics-related tweets 1,011 (4%) times. A manual annotation of 100 classifications found 7 false positives and 3 ambiguous locations, giving the model a precision of 0.9. Likewise, a manual annotation of 100 non-classifications found 1 false negative and 2 ambiguous tweets, giving the model a recall of 0.97. Therefore, the F1 score for the politics topic identification model is 0.93.

After filtering the model results, there were some interesting results, as shown in the tables below:

Name Gender	Count	Percent
Ambiguous	18659	66.6%
Male	5696	20.3%
Female	3682	13.1%
Total	**28037**	

Topic	Name Gender	Count	Percent
Neither	Male	5221	19.9%
	Female	3453	13.2%
	Total	26172	
Sports	Male	242	28.3%
	Female	73	8.5%
	Total	854	
Politics	Male	233	23.0%
	Female	155	15.3%
	Total	1011	

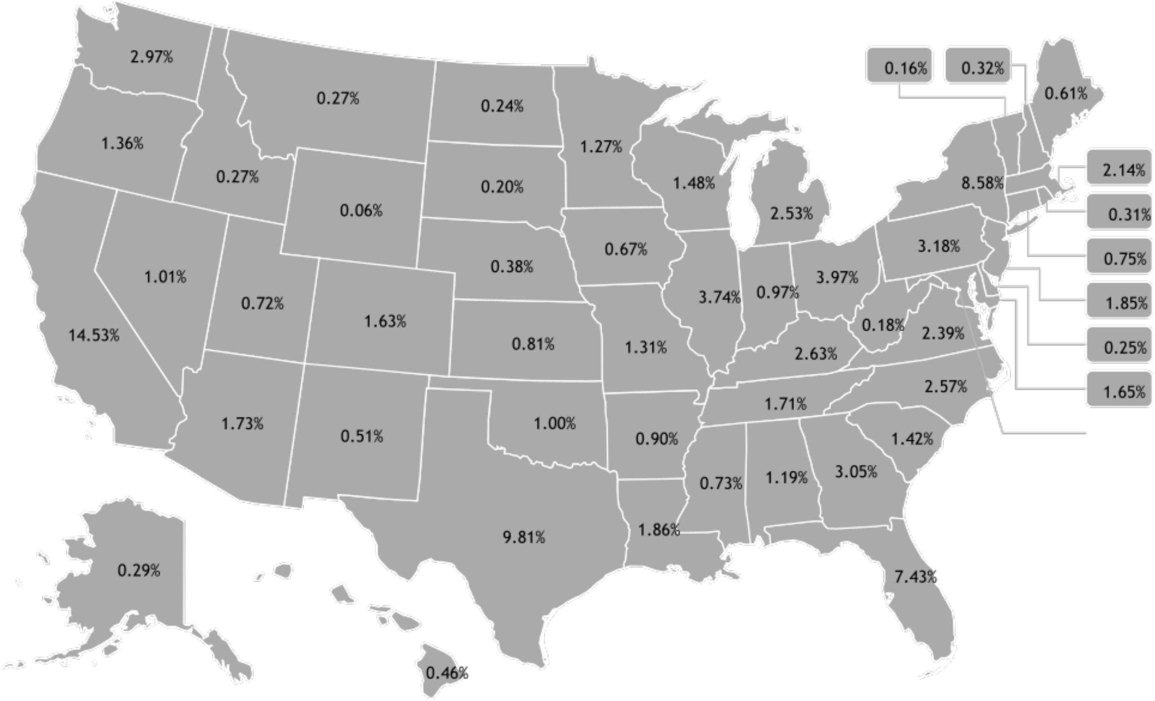
Name Gender	Topic	Count	Percent
Ambiguous	Sports	539	2.9%
	Politics	623	3.3%
	Total	18659	
Male	Sports	242	4.2%
	Politics	233	4.1%
	Total	5696	
Female	Sports	73	2.0%
	Politics	155	4.2%
	Total	3682	

First off, the percentage of male gendered names and female gendered names do not seem to be equal. This could be due to two reasons: 1. Twitter's userbase is not 50/50 male/female or 2. Female gendered names are more ambiguous and this model could not identify as many of those names.

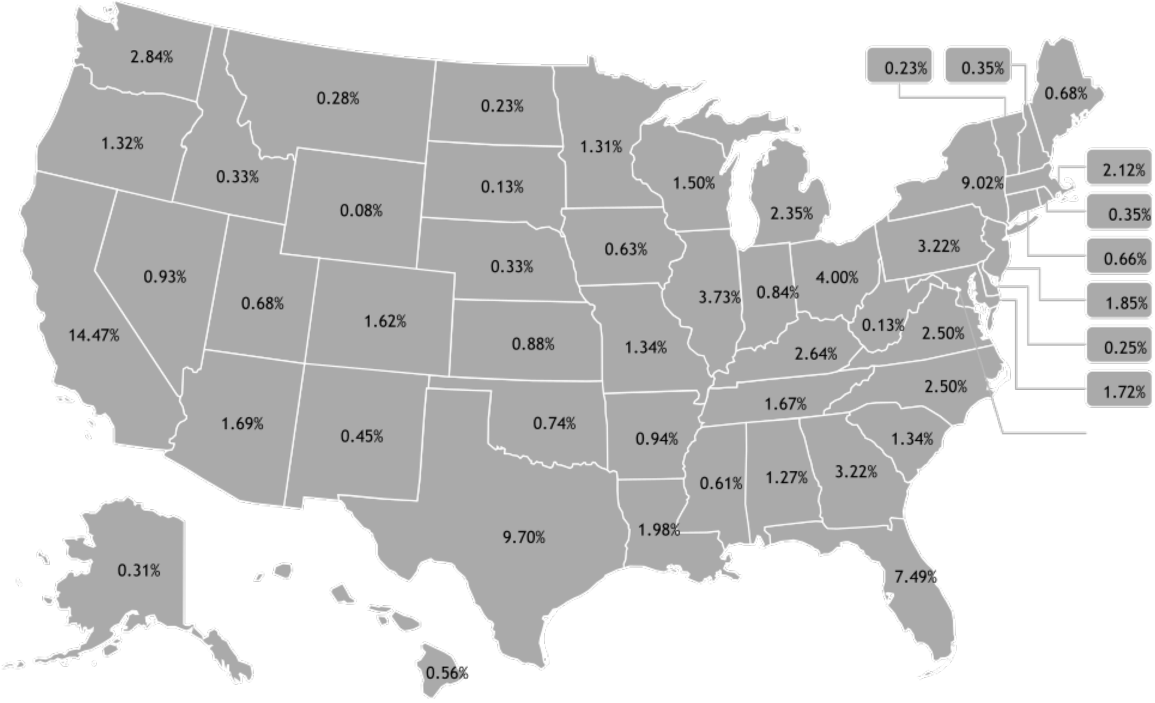
Another interesting observation is that for sports topics, Twitter users with male gendered names tend to tweet more than those with female gendered names. This difference does not seem to exist with politics-related tweets.

With these results, the following visualization "maps" were created:

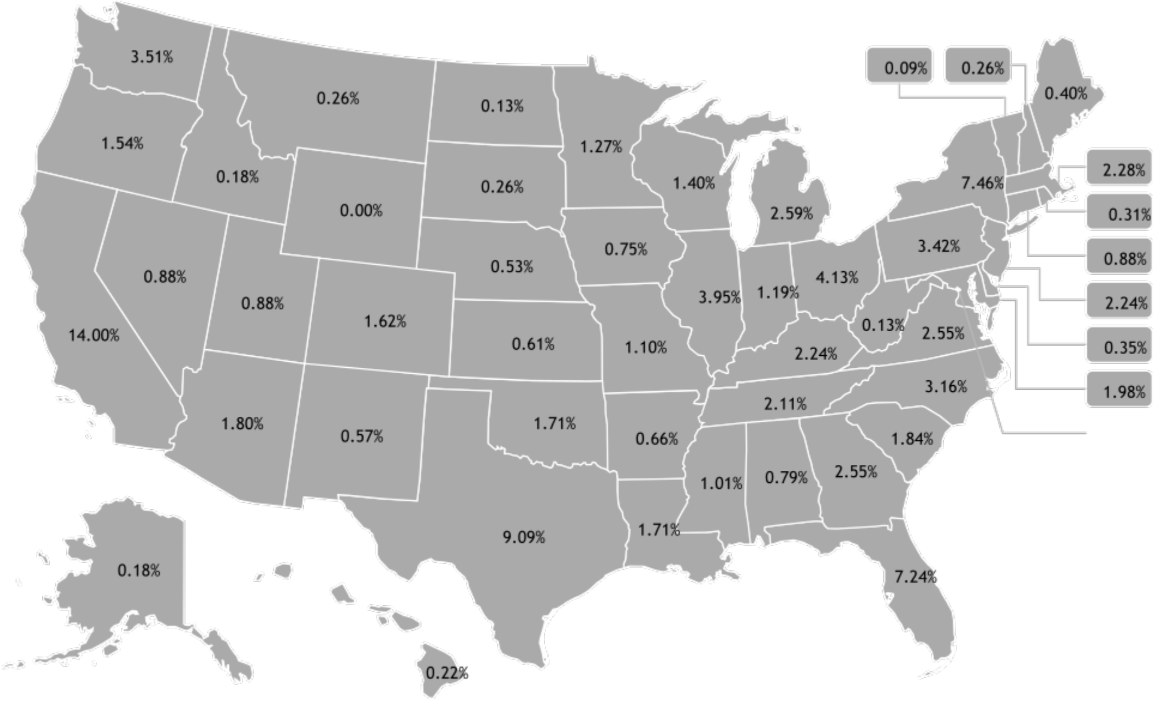
Percentage of Tweets by State



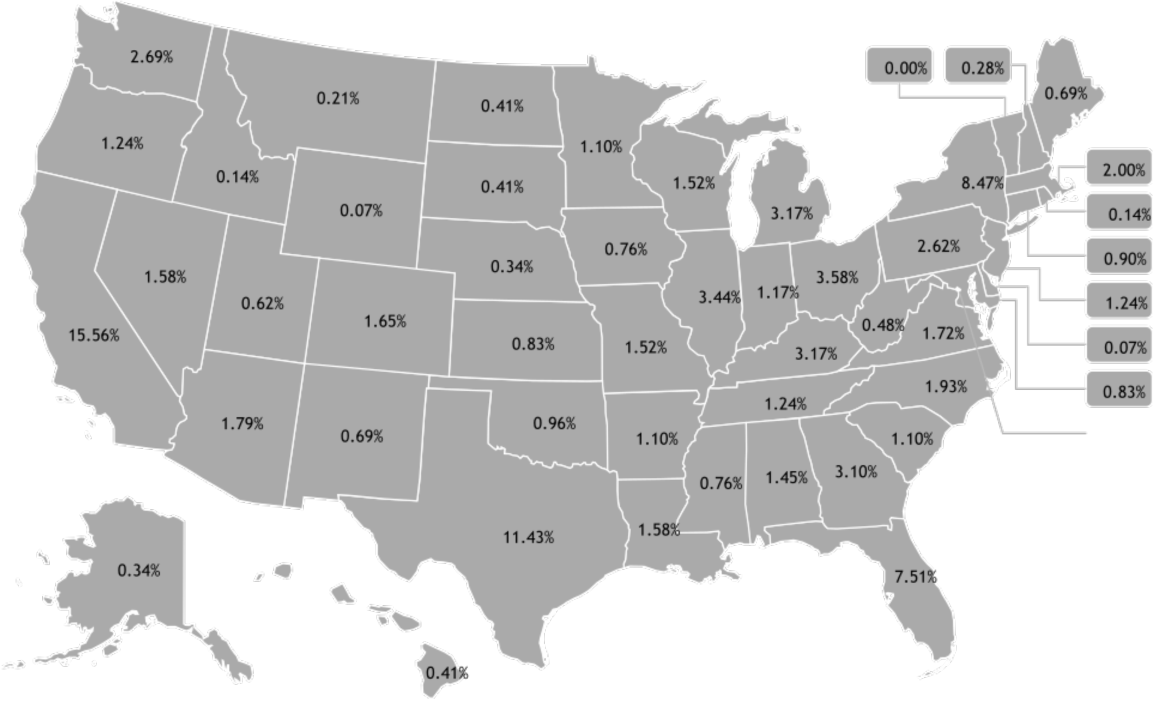
Percentage of Tweets from Ambiguous Gender Names by State



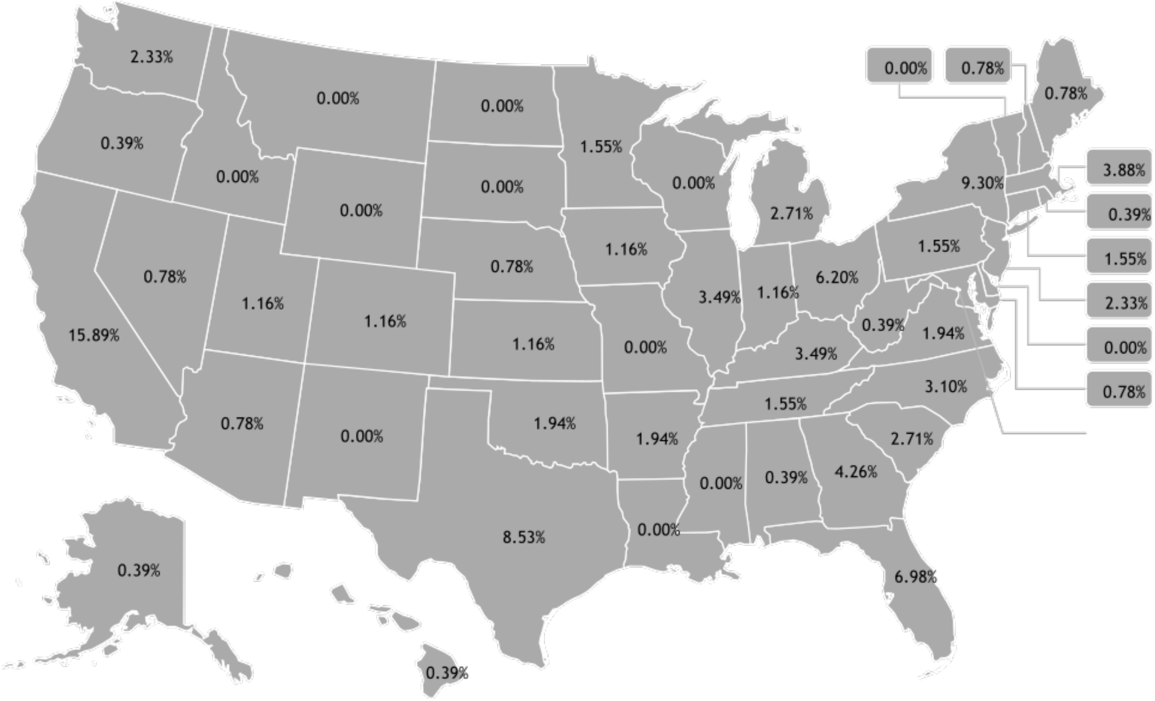
Percentage of Tweets from Male Gender Names by State



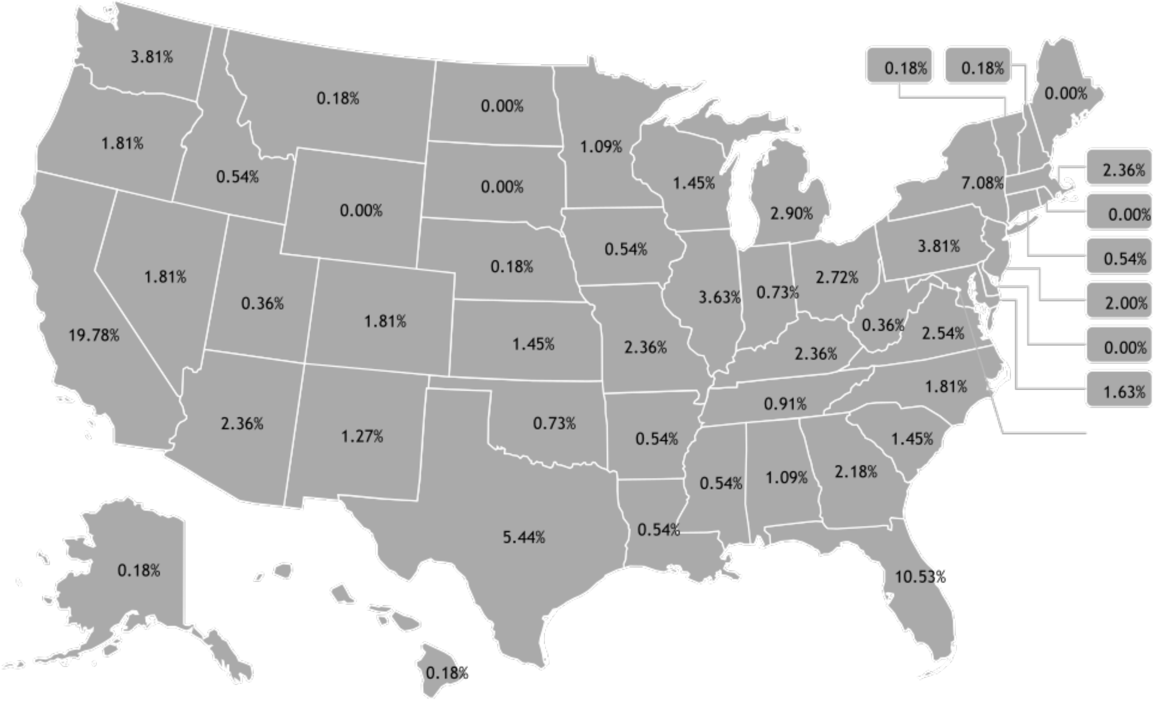
Percentage of Tweets from Female Gender Names by State



Percentage of Tweets about Sports by State



Percentage of Tweets about Politics by State



The top 10 states by percentage of tweets (where a state was identified) were:

1. California: 14.5%
2. Texas: 9.8%
3. New York: 8.6%
4. Florida: 7.4%
5. Ohio: 4.0%
6. Illinois: 3.7%
7. Pennsylvania: 3.2%
8. Georgia: 3.0%
9. Washington: 3.0%
10. Kentucky: 2.6%

Here are some interesting takeaways from these maps:

- There were relatively more female gendered name users from populous states such as California and Texas
- California and Florida tweeted relatively more often about politics whereas Texas and New York tweeted relatively less often about politics
- California, New York, and Ohio (maybe NBA playoffs-related) tweeted relatively more often about sports whereas Texas and Florida tweeted relatively less often about sports

Programs

There are three main Java files in the attached zip file along with two Python scripts:

- *TwitterUtilities.java*
- *SearchTwitterAndStoreTweetsInFile.java*
- *GeotargetFormat.py*
- *AnalyzeTweetsFile.java*
- *Results.py*

As you can see, these are rewrites (some minor, some extensive) of the tools we used in Assignment 2. I will explain each below.

TwitterUtilities.java

This file only has a one-line addition at line 125. It sets the count of the query to Twitter to 100 (the max) from the default value of 15. This allows me to pull 100 tweets for each query, giving me more data given Twitter API's rate limits.

SearchTwitterAndStoreTweetsInFile.java

The first change in this file from Assignment 2 is the query string. Instead of searching for specific text regarding companies, I search for "", which returns all tweets form Twitter.

Next, I made substantial changes to the *writeTweetsToFile* function.

1. I changed the name of the file output to include the timestamp so I could continuously write files (see below) with different file names.
2. I created three filters at this level before I even analyzed the tweet: the user has to be non-null, the location has to have a value, and the user's language has to be English. The first two filters are required so I always have user information and location to analyze for the user's gender and state, respectively. The last filter is self-explanatory, as I only want to analyze tweets written in English.
3. I outputted the file with the following fields separated by tabs: tweet text, user name, user location, and user screenname.

Lastly, in the main function, I wrote a timer around the *writeTweetsToFile* function that calls it every 65 seconds. I ran this for several hours to collect over ten thousand tweet data, and can collect more data if necessary.

GeotargetFormat.py

I wrote this simple Python script to reformat the state names CSV I got from Google Adwords API. This was required because there were some inconsistencies with the formatting of the raw file. It then outputs a *Newgeotargets.csv* for use in *AnalyzeTweetsFile.java*.

AnalyzeTweetsFile.java

This file, unsurprisingly, had the most deviations from the original Assignment 2 version. First, I defined several global string array variables that will be explained below:

- A list of all 50 U.S. states
- A list of abbreviations for all 50 U.S. states (in the same order as the states)
- An empty array of analysis results
- A list of politics-related words
- A list of sports-related words

I defined a new function *makeMap* that creates HashMaps for both states and baby names. It reads in CSV files and puts each row into a key value pair in the HashMap. I take a very conservative approach on collisions for both HashMaps:

- For states, the key is the city name and the value is the state name. If there is a key collision (if a city is found a second time), the original city is removed and that city is added to a *collisionList* so it is always ignored.
- For baby names, the key is the name and value is the gender. If there is a key collision (if a baby name is found a second time), the original baby name is only removed and added to a *collisionList* if the value (boy or girl) of that key is different. This is because the baby names file can have identical entries for the same name (Michael->Boy, and then Michael->Boy again) and I don't want to ignore those. However, if the same key yields different values (Alex->Boy, and then Alex->Girl), I want to ignore that baby name for our analysis.

processTweetFile is a new function (replaces *processTweetFileSentiment* in Assignment 2) that calls all of the relevant functions that actually perform the analysis on our list of tweets.

1. Call a function to convert both politics and sports word lists to lowercase.
2. Create both state and baby name HashMaps.
3. Read in each line of the tweet file and calls *AnalyzeTweetInformation* on it.

AnalyzeTweetInformation is a function that takes a single tweet from our input file and analyzes it in the following way:

1. Count the number of politics and sports-related words in the tweet text.
2. Call *getStateLocation* to get the state name from the location field.
3. Call *getNameGender* to get the gender of the name from the user name and screenname fields.
4. Print out the results to the console.
5. Add the resulting string to *tweetResults* for output later.

getStateLocation is probably the most involved function in this tool. Below is the step-by-step process it takes to find a state (if possible). Note that steps 3 through 7 were ordered intentionally from most accurate to least accurate to minimize errors.

1. Given that I already filtered for English tweets, I also needed to filter out some of the most common non-U.S. locations. I used this website (https://blog.cudoo.com/which-countries-have-the-most-english-speakers) to get an initial idea and then looked at errors in the results for more locations. With these locations, I created a regex and only proceeded to the next step if the location did not match any of these locations.
2. I created a HashMap with the key value pair of state abbreviations matching to state names with the global variables created above. I also added a couple of one-off matches that are necessarily state abbreviations but are common in the results (NYC and Cali matching to New York and California, respectively).
3. In this step, I begin the matching. First, I check for locations with a "City, State" format and attempt to match the state. In this case, the state "part" of the location field can both be an exact match of an abbreviation or it can contain the actual state name.
4. If the above step did not match a state name, I then check for locations with a "City State" format and attempt to match the state in the same way as step 3.
5. If the above steps did not match a state name, I then check to see if the location is just a "State". In this step, if the match returns "Washington", I make sure it doesn't also contain "DC" or "D.C", since we do not want to match on Washington D.C.
6. If the above steps did not match a state name, I then check the "City, State" and "City State" formats (as well as just "City") again, except this time on the city field using the HashMap.
7. Finally, this function returns "NotAState" if nothing was found, or the state name if it was found.

getNameGender looks at both the user name and screenname fields to try to find the gender of the name of the Twitter user using a HashMap of baby names.

1. I start with the user name since this is likely to have more accurate matches. I check for an exact match of the first part of the user name (first name or only name) to a key in our HashMap. I only check for names that are more than two characters long, since anything shorter has too much potential for error.
2. Only if I couldn't match a gender name to the user name do I consider using the screen name. For the screenname match, I have to use a contains match, so I only consider names in our HashMap that are longer than 4 characters to minimize errors. Also, there are many instances when I could get multiple screen name matches within one screen name (for example "jamiepeterson" matches "jamie" and "peter"). In these cases, I only make the match if all of the matches yield the same gender in order to be conservative (if not, I will set the result to "Ambiguous").
3. Since the name match is more accurate than the screenname match, I return that if a non-ambiguous value exists. If the name match is ambiguous, then I return whatever was matched in the screenname match.

At the end of the file, *writeTweetInformationToFile* takes the list of results in *tweetResults* and outputs it in CSV format to TweetResults.csv.

<u>Results.py</u>

This Python script performs all of the filtering and analysis after the Java programs above output the results to TweetResults.csv. First, it reads in the CSV file into a data frame and filters on the data that had successful analyses. Next, it outputs 100-line samples of each analysis for the manual annotation mentioned above. Then it prints out some more stats that are detailed in the above section. Lastly, it outputs SVG files that generated the maps shown above.

Action Items and Surprises

In terms of action items, the first one is to delve deeper into why this model did not return a 50/50 split of male and female gendered names. Is it because Twitter users are not evenly distributed? Does anyone even have this information since Twitter doesn't ask for gender when a user makes an account? Or is it an effect of the level of ambiguity among female names vs. male names?

Knowing which states are tweeting more about sports can often be intuitively deduced. For example, during the NBA finals, there will likely be more tweets from California and Ohio due to the Warriors and Cavaliers playing. However, with respect to politics, this model can provide more interesting results if run on a larger data set. It can answer questions such as: Which states tweet/care more about politics? Do male or female gendered names tweet more about politics in specific states? What is Twitter's relatively engagement with politics (percent of tweets about politics) over the last 1/7/30 days?

There were two main surprises that I encountered in the course of developing this tool and looking over its results. First, I was surprised by how many names are actually gender ambiguous. Names like Michael and Elizabeth were not necessarily *always* given to male and female babies, respectively. This along with the disassociation between gender and sex in today's society makes gender classification by name a more difficult NLP problem than I had initially anticipated.

Second, I was surprised at the relatively low percentage of tweets that were about either sports or politics (about 6-7% combined). Maybe this is due my lack of familiarity with the current Generation Z's relationship with Twitter, but a surprising number of tweets were of little substance and didn't really mean anything. When I read about tweets through sources that reach me on a daily basis, it seems to always be about sports or politics, but this is apparently a small subset of what's really out there.