Problem Motivation

There is a vast amount of available geotagged data from social media. About 0.5-1% of tweets from Twitter are geocoded. There are 340 million tweets per day, so this translated to 1.7-3.4 million geocoded tweets per day. We set out to investigate the question “is it possible to make accurate predictions about real-life events using this data?”. If it is possible, it would be beneficial for a number of reasons. Possible applications include natural disasters, political uprising, crowdsourcing, and geo-targeted advertising. We already know that the Red Cross uses geocoded tweets for crisis monitoring, to enable rapid response in these situations.

We ultimately decided to use geocoded tweets to predict the results of the U.S. presidential election, because it is culturally relevant, useful, and interesting to us.

Related Work, Contributions, and Novelty

Election prediction is a subject of great interest. Both candidates spent about 1 billion dollars this election, and billions more are at stake. The political future of the nation, and to a degree, the world, is affected by this event. There is daily statistical polls from a variety of organizations, and predictions are constantly presented in the 24-hour news cycle. These polls, however, are based on voting and not on social media data.

There have been a few studies on tweet classification. Matt Bush, Ivan Lee, and Tony Wu at Stanford University did a study of tweet classification with basic NLP (natural language processing) algorithms. However, they simply classified tweets into topic (e.g. personal, sports, news, etc.) A similar study was conducted at Ohio State University In this study, external domain-specific features such as the author's profile data was also used. Researchers at the University of Maryland have a project called TwitterStand, in which they tried to create a news aggregation service by processing tweets, as well as looking at the popularity of users. For example, a frequent followed user such as CNN would be highly weighted in this scheme. They tried to determine which tweets are breaking news. None of these studies have a spatial component. Hanan Samet (also at University of Maryland) further developed this work with a spatial component. He has a project called NewsStand. This project incorporates a spatial component into the news, but did not use Twitter data.

So, while there have been many studies on election prediction, tweet classification, and spatially targeted news, there has not been a study combining geocoded tweet data for election prediction. Our project contains all of these elements. In addition, the previous tweet classification studies labeled tweets as different types of news, such as sports, politics, etc. We instead aim to determine not only which candidates are being mentioned in tweets, but whether they are in favor of or in opposition to the particular candidate.

Problem Statement

The focus of this study is to examine the questions “Can we predict how counties will vote in the US Presidential election by classifying the geocoded tweets from those counties?” and more
specifically, “Does the aggregate of tweets mentioning a U.S. presidential candidate's name correctly predict how a county will vote?”. We chose to examine this at a county level because we have the polygons for counties, election results are released for each candidate, and they have a sufficiently fine granularity. We decided to use Twitter because of its high volume and velocity of data, its geocodes, and the existing APIs for Twitter.

The inputs are the incoming tweets from Twitter stream, each of which can be thought of abstractly as a vector of word counts. The outputs are a prediction of which candidate will win each county.

**Challenges**

There are a number of issues that make this project difficult. A relatively low proportion of tweets are geotagged (~0.5%-1%), and we are sampling a subset of those. This still proved to be ample data, but many of these are incorrectly geotagged. We are limited to an extent by our computers. We did not have access to advanced computing resources for this assignment, so this limited the sophistication of our solution to an extent.

Another main issue is a spatial one, and it has to do with visualization. It can be difficult to compare point data from Twitter to external geographic data sources, and difficult to visualize tweet data. This will be discussed in more detail in the “Preliminary Results” section.

**Proposed Solution and Methodology**

In this section we give a high level overview of our proposed solution, followed by a more in-depth description of the tools we used as well how we used them.

Our data files, database schema, scripts, etc. can be found at:

[GitHub for CSCI 5980 Group 11](https://github.com/CSCI5980Group11)

First, we needed to collect our data. We harvested tweet data from important milestones in the election. We chose to use the vice presidential debate, the final presidential debate, and the day before as well as the day of the election. We then stored this data logically in a database. Next, we classified each tweet as pro-Obama, pro-Romney, anti-Obama, anti-Romney, or neutral. We then we aggregated our results by county. Finally, we obtained ground truth data from the election and used this data to validate our results.

**Software Stack:**

We are using some of the standard tools for dealing with spatial data to complete this project. These tools are all free and open source so it is easy for others to replicate and add to the work that we have done. For people that are familiar with the software packages that are mentioned below feel free to skip this section on go straight to our methods. For people that are not familiar with these packages some of the links below should serve as a starting point to understand some of the technologies that we are using.
Tweepy for Twitter API - This is an easy package to use, but has almost no documentation. It is capable of easy and full access to the Twitter API, but it sometimes involves looking through the source code because of the lack of documentation. There are many options for accessing the Twitter api inside of your favorite scripting language (Twitter).

SpatiaLite for spatial database - This class has focused on using Oracle spatial, but we thought that it would be fun to try out another spatial database. SpatiaLite is small, lightweight, and easy to use. It allows for almost complete sql inside of an embedded database which makes it easier to setup and manage databases. There are clearly limitations in terms of performance in comparison to oracle but we thought that it would be nice to learn an open source package for advanced spatial databases. SpatiaLite databases can be viewed and queried inside of QGIS, and also ArcGIS making them very usable from a mapping perspective. I do not know how the spatial components stack up, but SQLite tests faster than MySQL and PostGIS (http://www.sqlite.org/speed.html). All of the concepts for spatial operators seem to match up nicely with the oracle spatial work that we have done. For those that want to learn more here are some links

tutorial: http://www.gaia-gis.it/gaia-sins/spatialite-tutorial-2.3.1.html

QGIS for visualization - Quantum GIS is a simple, free, open source GIS package that allows for most of the functionality found in other GIS packages. It can run on Linux, Mac, and PC and offers the ability to make some great maps. More information on QGIS can be found at http://www.qgis.org/. QGIS is a very intuitive way to look at a lot of geographic data on any computer. It is not capable of all of the things that ArcGIS can do, but for this project it works great. Also for our team we prefer to not work on Windows, so this is a nice touch.

Python - Python is serving as the glue language in this project. Twitter objects from the tweepy API are parsed using Python, and put into the spatial database. Complex database queries are conducted using the pyspatialite add-on for Python. Our clustering and other advanced processing is also done using python packages. Obviously there are many other ways that this work could be done, but the ease of python, and the amount of plugins make it a good choice. Python is supported in SpatiaLite, and in QGIS, so Python does make a lot of sense to use here. Some of the modules that are being used are

a. scipy for cluster analysis
b. numpy to store matrices
c. pysal to work between different formats and for spatial analysis
d. pyspatialite to work with the database
e. tweepy - which has already been talked about

For information on general python here are some links

Google tutorial: http://code.google.com/edu/languages/google-python-class/
Python tutorial: http://docs.python.org/2/tutorial/
A note regarding documentation: documentation is very good for the core of Python but is very weak for the SpatiaLite and Tweepy modules.

Methods

The basic process that we used here was to create a Twitter stream listener that dumps data into a spatiaLite database. This allowed us access to gigabytes of geo-tagged data without much computing power. The database used allows for visualization in QGIS (and with some more work the web) so this method allows for an easy way to look at data quickly. The rest of this section will look at depth at how we created the TwitterListener and the spatial database. This will be technical and specific to the software choices that we made.

Twitter listener

Tweepy has a built in module to listen to a Twitter stream with ease. There is an example in the documentation shown [here](#). We searched for the following political terms: "debate", "Romney", "Obama", "Biden", and "Ryan". We changed the listener to dump data straight to our database rather than printing the results to the standard output, and to take arguments for search terms (this does not seem to really work as expected).

The TwitterListener API allows for input in keywords and locations of interest. Our early streams allowed for all tweets that had a location [-180, -90, 180, 90], but as we focused more on the election it made sense to focus the minimum bounding box on the united states. Even with this bounding box in place there were many tweet that were received that were well outside of the bounding box. This seems like an error either in the Twitter API or the tweepy module. For the keyword input tweepy takes a list and feeds it to Twitter. This does not seem to limit the stream to just tweets that have that keyword, so it is sort of unclear how to filter tweets with this method.

The Twitter API and tweepy also allow for a searching API for searching for tweets with specific content or from specific users. This method is limited to 350 requests per hour so we did not use it for this project, but it could be useful for various applications, or as an add on to this project.

Twitter parser

The Twitter parser script takes a single Twitter status (a tweet) object and places the information from it into the correct area of the database. The script currently only processes tweets that have geographic information. This is about 1% of the total stream, and about 10% of the locational stream that we had open. By dumping the data straight into the database we have the issue of losing some of the data (data not important enough to put in our database) but the speed gain of not having to save the raw data and then place the data into the database. The biggest challenge that we faced here was figuring out the syntax to put the geographic data into the geographically aware column in the spatialite database. The syntax for this is not clear based off of the documentation, and it is a little bit trickier than it should be.

Schema for database
After a little bit of thought we designed and implemented the following schema to hold the Twitter data. The high level overview shown below:

status - holds the information for each tweet - linked to user table on user id (*spatial)
user - holds the information about a user
mention - links to tweet of mention and user of mention
url - stores each url from each tweet
word - each word in each tweet (primary key on tweet and word number)
hashtag - each hashtag in each tweet (primary key on tweet and hashtag)

The most important table in the database is the status table, because it contains both the location and text of the tweets. There is important information listed about each user in the user table that can easily be linked to tweets and mentions. The mention, url, and hashtag tables are actually subsets of the word table. All of these tables extract specific words out of each tweet and save them together to allow us easy access to the information of the tweets. Mentions are slightly different than the other tables listed above because it also is related to the user table. We created the place table based off of information from the Twitter API. Our schema file can be found online with our other project files.

Classifying Tweets
We broke each tweet into a vector of words. For each tweet, we iterate through this vector of words twice. In the first pass, we determine if the tweet is associated with Obama, Romney or neither. This is done by checking for presence of words associated with each respective candidate. Example query:

```
select lower(tweet) from ___ where lower(tweet) like "%obama%"
```

If the tweet contains more instances of the word “Obama” than “Romney”, it is marked as an “Obama” tweet, and vice versa. The tweet is marked as neutral if there are equal occurrences.

In the second pass, we determine if tweet is in favor, in opposition, or neutral to the candidate based on the presence of negative or positive words (we created a list of these words). We assigned a positive score to the tweet if it contained more positive words (or an equal number) and a negative score to the tweet if it contained more negative words. So, the relevant candidate would have their score increased if they were mentioned in a neutral or positive context, and decreased if they were mentioned in a negative context. These scores were aggregated per county to make a prediction for the county.

Preliminary Results
This section is based on data up until the day of the election. The final validation and a more in-depth analysis can be found in the following section.

A Twitter stream will provide data until it is closed. With our streams leaving one open for a few hours would create hundreds of thousands of geolocated tweets. This made it very difficult for us to use the data inside (because of the sheer quantity).
The map above shows all tweets which mention Obama as blue dots, tweets which mention Romney as red dots, and all other tweets as white dots. These are tweets from the night before the election. We decided to aggregate the data by county so that the data could be compared easily with data from the census and other sources, and so that the data could be easily visualized.

The process of assigning each tweet to its county is a time-consuming one. Each of the (over 3000) counties in the United States needs to look at all of the tweets and figure out which tweets are inside of its borders. (Note that none of the computation is repeated in this process). With a simple county map being pretty detailed and complex, and with hundreds of thousands of tweets the process is pretty slow. Even with an R-tree index (which is placed on the spatial files) this process can take hours.
In the above map, those counties which are grey are split between Obama or Romney. In some cases this is because there are 0 political tweets from the particularly county, which trivially creates a split. An interesting fact to note from the map above is that swing states like Florida, Ohio, and Pennsylvania clearly show less beige counties than most other states. This might be an indication that they are more engaged (via Twitter) in politics.
The map above from the New York Times shows the election results by county. There is a strong visual correlation between the two images.

Tweets vs population by county
We see a clear positive relationship between population (x-axis) and tweets per county (y-axis). This relationship is intuitive, but it is nonetheless beneficial to verify it. This also shows that we are able to compare our data with other geographic data. Clearly areas with higher population should have more tweets, but how do counties deviate from this trend? The simplest way that I could think of showing this was to map the number of tweets per person by county. Counties with very few tweets per person are shown as white and the counties with a lot of tweets per person are shown in red. It is pretty clear that the east coast tweets more per person than the west coast. We can clearly see that there are more people sending more tweets in the east than in the west. There are over 400,000 tweets and roughly 3000 counties in the U.S., so we find it unlikely that we have too small of a sample size. However, our sample is for some counties is very small, and thus can easily be skewed by one or a few individuals.

**Final Results, Analysis, and Validation**

We clearly wanted to make some sort of comparison between the Twitter data by county and the how each county voted. Based on the preliminary results and visual analysis discussed earlier, it is difficult to discuss the nature of this relationship at a more in depth level without actually knowing how each county voted. To obtain this data, we used the website politico.com, which has a page for each state with results for each county. To parse the data from the 50 states we wrote a script that would download them all and then rip the data out of the tables.
into one table. This is not really part of our project, but because of the time that we were forced to spend on it, this part does seem to warrant some discussion. (Furthermore, it would be useful for someone trying to reproduce these results or to apply these techniques in another situation.)

To gather validation data for each state, we implemented a loop which iterates through each state and called wget to download the results for all counties for a given state. Then, we used a Python package for HTML scraping, called BeautifulSoup, to find each row in the table and dumped the information for Romney and Obama into a table. More information for this parsing can be found by following the beautifulsoup link above.

**State Results**

Maps presenting our predictions (Tweet winner by state) and the validation data (Vote winner by state) are shown below.
We made predictions for 48 states: specifically, all 48 of the contiguous states, excluding Montana, for which we had insufficient data to make a confident prediction, but including District of Columbia. Of these states, we correctly predicted 32, so our accuracy is 66.66% (2/3). We attribute a majority of the inaccurate predictions to sampling bias from Twitter. A confusion matrix is presented below to further analyze these results.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Obama</th>
<th>Romney</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obama</td>
<td>21</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Romney</td>
<td>11</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Obama Precision</td>
<td>21/32 = 65.63%</td>
</tr>
<tr>
<td>Obama Recall</td>
<td>21/26 = 80.77%</td>
</tr>
<tr>
<td>Romney Precision</td>
<td>11/16 = 68.75%</td>
</tr>
<tr>
<td>Romney Recall</td>
<td>11/22 = 50.00%</td>
</tr>
</tbody>
</table>
We find the precision and recall analysis to be rather illuminating. The most telling measure above is the Romney recall: we only predicted half of the states that Romney would win. Furthermore, we predicted Obama to win over 80% of the states that he ended up winning. These facts indicate that there is more Twitter support for Obama than actual voter support. In other words, Twitter is giving us a biased sample of U.S. voters.

**County Results**
The following maps show the tweet winner by county and the vote winner by county, respectively.
These maps tell a similar story. We tend to slightly overestimate the voter support for Obama. More precisely, 2158 out of 3109 counties were correctly predicted, for an accuracy of 69.41%. One interesting fact to take notice of is that the majority of incorrectly predicted counties have a very low population, which can lead to two possible conclusions. First, we should maybe institute a higher threshold for how many tweets a county should have before we make a prediction. Second, and perhaps more interesting, is the possibility that Obama supporters are a more outspoken (via Twitter) in counties where they may be the minority. Or, it might not be simply a characteristic of Obama supporters, but rather regional political minorities in general. Of course, there is also the aforementioned sampling bias of Twitter that is undoubtedly at play here once again.

The following maps showing the percentage of tweets as well as votes for each candidate by county further drives home this idea.
Nationwide Popular Vote
One final surprising and exciting result is that, after aggregating all tweets, we predicted 52.0% support for Barack Obama and 48.0% support for Mitt Romney, while the U.S. popular vote yielded 51.0% support for Obama, 47.3% support for Romney, and 1.7% for the other candidates. So, our prediction was within 1% of the results. Also, take note that we did not classify tweets for third party candidates such as Jill Stein or Gary Johnson.

Conclusions and Future Directions

Overall, the results we obtained were pretty accurate. We were able to predict nearly 70% of all U.S. states and counties correctly, and our aggregate nationwide prediction was within 1% of the popular vote. We think that these results are greatly affected by the large sampling bias of Twitter. The following chart (from an external source) highlights this quite vividly.

An overwhelming majority of Twitter users are between 15 and 25 years old. In this past election, Barack Obama won 60% of the vote of those between 18 and 29 years old. This explains the high Obama recall and low Romney recall that we found while analyzing the state predictions and results. It may be the case that there is not much room for improvement for data with this amount of bias, at least with using raw Twitter streams.

Some ideas for future directions to take this work are as follows. First, it can only help to gather more data. A longer period of tweet collection would be beneficial. This would better equip us to support more sophisticated spatio-temporal analysis. It would also be worthwhile to investigate
how campaign events can be linked to changes in the Twitter support in different geographical locations (e.g. a candidate visiting a particular county). Some of our peers suggested that we supplement the Twitter data with different social media data. For example, Facebook supports geolocation, has approximately twice as many users as Twitter, and may be a better representation of population. More sophisticated (and robust) algorithms for detecting intent can be instituted. These might not have been necessary for a problem such as the one we tackled in this project, but there are numerous other events and phenomena for which better NLP algorithms may be crucial. Staying within the realm of politics, our methodology can even be used for other elections for which there is a lack of available polling data, such as less prominent U.S. races, or elections in other regions of the world.