Project 3- CSC 113 Introduction to MATLAB, Fall 2014

Twitter Trends

In this project, you will be developing a geographic visualization of some twitter data from across the USA. As an example, consider the following map which portrays how people across the country feel about Justin Bieber (based upon an analysis of their tweets). States that are red have the most positive view, while states that are dark blue have the most negative view; yellow represents a more neutral view, while states in gray have insufficient data.

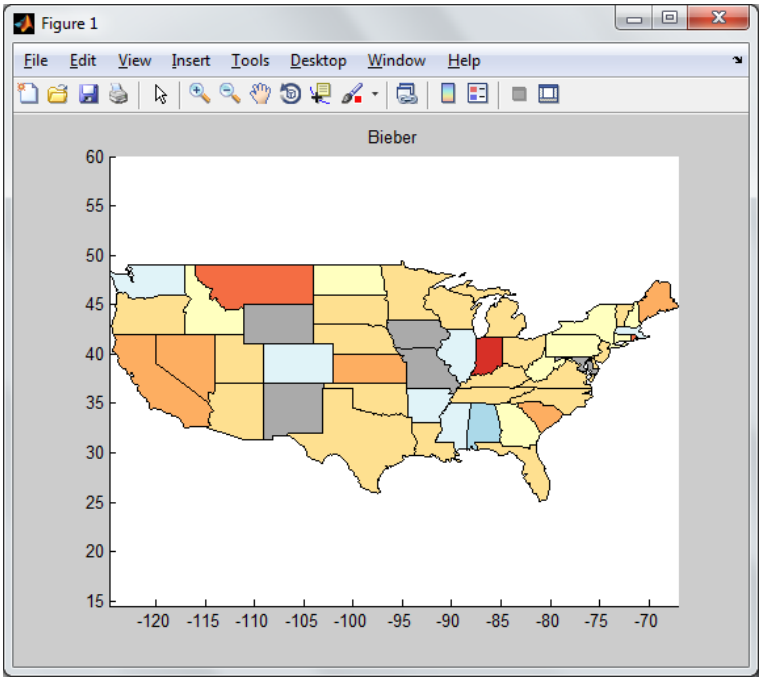

Figure 1

The methodology used to produce this figure is as follows:

1. An input file is provided that has Twitter data, where each tweet contains the word 'bieber'.
2. Each tweet comes with geographic meta data in the form of latitude and longitude. That can be used to associate the tweet with a specific state.
3. The "sentiment" of a tweet is determined by breaking the tweet message into individual words, and then looking up each word in a sentiment dictionary that maps certain words to floating-point values in the range [-1, +1]. A sentiment of -1.0 is the most negative, while a sentiment of +1.0 is the most positive. We will rely upon an existing classification of about 18700 words with sentiment values. For example:
   ' demoralizing' receives a score of -0.625
   'bad' receives a score of -1.0
   'good' receives a score of +0.95
   'excellent' receives a score of +1.0
   If a word of the tweet is not found in the sentiment dictionary, it is ignored. The overall sentiment of the tweet is the average of the sentiment scores of all the words that are found in the sentiment dictionary. If no sentiment scores are found for any of the words of the tweet, this tweet is ignored.
4. The overall sentiment of a state is computed as the average sentiment score for all tweets that are associated with that state (ignoring those tweets that did not have a sentiment score). The state's sentiment score is then mapped to a color between blue (negative) and red (positive) using a prescribed color gradient.

For this assignment, you will be able (but not required) to work in groups of two. Partners for this project are to be chosen independently; should you want to work in a group but are not able to find a partner, you are free to use the forums to find a partner for your project. Please note that you must choose a different group member than those you collaborated with on your previous projects and must choose other group members for subsequent projects.

1. Project Requirements

1.1. At Startup

At startup the script asks the user which input tweet data to analyze and that is done via a menu:

1.2. Input Files

After, the user makes a selection the appropriate input files are loaded into Matlab. The following input files are provided:

File name	Description

states.mat	This file contains a 1x52 Structure Array called **states** that has all the latitude and longitude data needed for plotting the boundaries of each US state. It contains data about all 50 US states plus D.C. and Puerto Rico.
	
	For example, the 23rd element contains the data for the state of MI:
	`K>> states(23)`
	`ans =`
	`name: 'MI'`
	`region: {[1x64 struct] [1x5 struct] [1x76 struct] [1x7 struct]}`
	
	Note that each element has two fields: a field called **name** and another field called **region.** MI has an irregular shape and thus it takes 4 different regions to describe its boundaries.
	
	The longitude and latitude of the first point of the MI boundary in its second region can be accessed by:
	`K>> states(23).region{2}(1)`
	`ans =`
	`longitude: 45.7305`
	`latitude: -85.5081`
	Note that for most of the states; only one region is needed to describe the boundaries.

centroids.xlsx	This file contains the name of a state and its centroid location. Informally, the centroid is an "average" of all geo positions in the state. You will use this single centroid position as an approximation for the entire state when determining the state to which a tweet is closest.
sentiments.xlsx	Contains a mapping of words and their assigned sentiment values between [-1, 1]
bieber.txt bacon.txt cat.txt dog.txt	Input files that contain the Twitter data. In the dog.txt for example each tweet contains the word 'dog' .

1.3. Analyzing the twitter data

Once the twitter data is loaded then analyze it:

- Loop through the list of tweets and for each tweet:
  - Compute the average sentiment for that tweet by breaking the full message into words, and looking up each word in the sentiment dictionary (sentiments.xlsx).

    You are to use the following rule for determining the words of a tweet message. The full message is broken into words based on white spaces. For example, if the original tweet message is:

    ```
    justin Bieber ...doesn't deserve the award.. eminem deserves it.
    ```

    The words of this tweet message are:

    ```
    ['justin','bieber','…doesn't','deserve','the','award..','eminem','deserves', 'it.']
    ```

Note that the word 'Bieber' and 'bieber' are considered the same and thus have the same sentiment score (i.e. your script must be case insensitive).

- Assuming the tweet has a sentiment score (that is, at least one word of the tweet was identified in the sentiments dictionary), assign this tweet's sentiment score to the "closest" state (using **`findStateOfTweet`**).

  Assign a tweet to the state that has its centroid closest to the location of the tweet. This is an imperfect rule (for example, tweets from New York City will actually be closer to the centroid of Connecticut and New Jersey then to the centroid of New York state); but it is an easy rule to implement.

- Once you have scored all tweets and assigned their scores to the appropriate state, compute the cumulative sentiment for each state as the average of all sentiments that were assigned. Then use that average state sentiment to pick an appropriate color for the state (using **`getColor`**).
- Finally, your script will have to generate a visualization as the one shown in Figure 1 (using **`plotMoods`**).

2. Design

Your main script must be called **tweetAnalyzer.m** and in there you ask the user to pick the Twitter data and analyze the tweets. In addition to the main script **tweetAnalyzer.m**, you are required to write and utilize the following user-defined functions.

Description of the User Defined Functions

```
function state = findStateOfTweet( tweetLoc, centroidsData )
%Assigns a tweet to the state that has its centroid closest to the
%geo location of the tweet using  haversine formula.
%Inputs:
%  tweetLoc: a 1x2 vector [latitude, longitude]
%  centroidsData: cell array that contains ALL the data from centroids.xlsx
%Returns:
%  state: a string that has the state abbreviation, e.g. 'NC' for North Carolina
```

Example run of the this function:

```
K>> findStateOfTweet( [36.1197, -115.1731], centroidsData )
ans =
NV
```

Note that geographic positions are spherical (even if we project them to two-dimensions for drawing a map). In particular, this means that when computing the distance between two positions, you cannot rely on the familiar planar equation. Instead, we will use the "haversine" formula (http://en.wikipedia.org/wiki/Haversine_formula). You are provided with a user defined function called **`haversine.m`** that you can just use to determine the distance between two locations. No need to understand/change the code in that file. All you have to do is just figure out how to use it.

```
function plotMoods(states, avgMoodState, dataFileName)
%creates a visualization of the US, where each state is assigned a color
%based on its average mood sentiment.  It does NOT show AK, HI and PR!
%Inputs: states: structure array provided in states.mat
%        avgMoodState: 1x52 array with avg mood b/n [-1,1] for each state
%        dataFileName: a string that contains the name of the data
```

This function is provided for you. However, it uses the **`getColor`** function so you must write that first.
The visualizations for all the input Twitter data are also provided as **`.png`** files so you can compare your visualization results. The visualization for the bieber.txt file is shown in Figure 1.

```
function color = getColor( mood )
%Assigns a color to a mood.  The colors are chosen
%based on Cynthia Brewer's Color Brewer (colorbrewer2.com)
%Inputs: mood is a real number between [-1,1]
%Returns: if mood is in the range [-1,1] a 1x3 color value from SENTIMENT_COLORS
%         otherwise it returns GRAY

SENTIMENT_COLORS = { [0.1922 0.2118 0.5843], ...
    [0.2706 0.4588 0.7059], ...
    [0.4549 0.6784 0.8196], ...
    [0.6706 0.8510 0.9137], ...
    [0.8784 0.9529 0.9725], ...
    [1.0000 1.0000 0.7490], ...
    [0.9961 0.8784 0.5647],...
    [0.9922 0.6824 0.3804], ...
    [0.9569 0.4275 0.2627], ...
    [0.8431 0.1882 0.1529], ...
    [0.6471 0      0.1490] };

GRAY = [0.6667, 0.6667, 0.6667];
```

Example runs of this function:

```
>> getColor(-1)
ans =
    0.1922    0.2118    0.5843
>> getColor(1)
ans =
    0.6471         0    0.1490
>> getColor(0.5)
ans =
    0.9569    0.4275    0.2627
>> getColor(5)
ans =
    0.6667    0.6667    0.6667
```

Notice that when mood (i.e. sentiment score) is outside the range [-1,1] then the RGB triple for GRAY is returned.

You can define any additional functions that help you organize your code in a more clear and modular fashion.

Matlab Built-In Commands that might be useful for this project: **`ismember`**, **`find`**, **`isempty`**, **`str2num`**, **`textscan`**, **`lower`**

Acknowledgment: this project is a variant of one developed by Aditi Muralidharan, John DeNero, and Hamilton Nguyen,

3. Styling Directions:

At the top of each of your **m-files**, add the following information:

```
% Name (s)
% Date
% Lab Section #
% Project 3:  TweetAnalyzer
```

Make sure that you suppress all *unnecessary* output.

4. Submission

This project will be **due November 24, 2014, at 2:00pm (one hour before the start of lecture).**

Before you submit your work, make sure the program:

- behaves as specified in this document. Consider what will be the output of each step.
- is thoroughly tested.
- satisfies the grade sheet for this project

4.1. Online Submission.

You need to submit your files through the course's Moodle site**. Zip up ALL the m-files used by your project in a folder called project3.zip**. You will need to submit at least the **three** required files (**tweetAnalyzer.m, findStateOfTweet.m**, **getColor.m)**. If you have written other user-defined functions then you will need to submit those as well. Zip up all files. The name of the assignment in Moodle is **Project 3 Submission.**

4.2. Hard Copy Submission – due at the beginning of lecture on November 24, 2014.

Print a hard copy of only the m-files that you have written. The hard copies will be collected at the beginning of lecture. If this is late (not ready when collected), you will receive a -5 pt. grade deduction.

- Print the grading sheet (last page of this document) and fill in your name(s), NCSU email, and Lab Section number.
- **Staple** your grade sheet on top followed by your printed m-files.

4.3. Extra Credit

To get extra credit you can submit the project early.

- Submissions by Nov 10, 2014 receive 5 points extra credit
- Submissions by Nov 17, 2014 receive 3 points extra credit

Grade Sheet Project 3: Twitter Trends, Fall 2014

Student 1:
Name : _____
NCSU Email: _____ Lab Section #: _____

Student 2:
Name : _____
NCSU Email: _____ Lab Section #: _____

Points	Earned	Description
Comments:		
1		Name(s), date, section, etc. in project header of each .m file
1		Correctly named files
3		Code is well-commented
3		Suppressed unnecessary output
Setup/Input Data		
4		Displays menu for which data to use
4		loads up the correct input files
tweetAnalyzer.m		
20		Correctly breaking up the twitter data into words
5		Correctly assigns a sentiment score to a word by using the provided dictionary
10		Correctly determines the average sentiment for a tweet
10		Correctly determines the average sentiment for all states
5		Correctly uses **findStateOfTweet.m**
4		Correctly uses **plotMoods.m**
findStateOfTweet.m		
15		Correctly determines what state to assign to a tweet
5		Correctly uses the provided **haversine.m** function
getColor.m		
10		Correctly determines a color based on a sentiment score

Points	Earned/ Deducted	Description
-5		Hardcopy turned in late, not stapled, or grade sheet improperly filled out.
5		Early submission by 11/10/14
3		Early submission by 11/17/14

Total Points Possible	Total Points Earned
108	

Graded by: _____

Date Code Submitted on Moodle : _____