SI 671/721: Homework 3: Spatial Data Mining

Due: Monday Dec 10th, 2018 by 11:59pm

Can be turned in up to Wednesday December 12th at 11:59 with no penalty

Version 1.1 (Last Updated: Nov.27th, 2018)

Summary

Different geographic regions each have their own way of saying things. For some, footware is called “sneakers” but for others it’s “tennis shoes.” When a region varies consistently in which words its residents choose, this is called a dialect. Spatial data mining can help reveal these regional dialects by finding which words are used more frequently in a region using statistical analysis. In Homework 3, you’ll replicate some of the maps we’ve seen in class for the US using Canadian tweets!

Task

We’ve done much of the prep work for you on this and have provided

- Word counts aggregated into Canadian census regions (approximately like US counties)
- A spatial distance matrix that describes which counties are considered “next to” each other.[1]
- Code for plotting the census regions where all you need to provide is a file with the census region’s ID and its scalar value.

Your tasks will be to

- Compute the local auto-correlations (Local Getis-Ord G*) for each word in the data
- Plot a few maps for individual words to show a range of regional and non-regional words
- Compute the SVD of a matrix where census regions

[1] In class, we talked about this matrix as “W” which could be computed using the “Rook” or “Queen” rules to identify which region are considered next to. In Homework 3, we’ll be following the same set up as Grieve (2011) where they define a county as next to another if it’s within 500 miles. While this is a broad definition, it lets us count most cities in the same Canadian province as co-varying in their language usage.

Learning Objectives

Homework 3 has the following learning objectives:

- Working knowledge of spatial libraries
- Familiarity with customizing map plotting libraries
- Experience with using the SVD

Data and Code

These files are provided:

- `subdivision_to_word_counts.tsv` is a tab-separated file where the first column is the Canadian census subdivision ID and the second column is word counts for within that region. The word counts in the second column are space separated, it's a word, followed by its count. Here's an example of the start of the first line in the file:
  `5915022 by 10610 david 212 featuring 786 is 18567 [...]`
  Here, the word "by" occurs 10,610 in subregion 5915022
- `gcsd000b11a_e.zip` is a zip file containing the shapefile and its metadata for all Canadian census subdivisions. The only thing you need to do with this file is unzip it and make sure it's in the same subdirectory as the Jupyter notebook you'll use for plotting
- `Plot Canadian Subdivisions.ipynb` is a Jupyter notebook you can use to plot everything. Plotting requires that you have `geopandas` installed. Right now, the notebook is set up to plot the colors using the file `random-subdivision-values.tsv` (described next). You'll need to change this file name to whatever your program outputs.
- `plot_canadian_subdivisions.py` is a python script that does the same thing as the Jupyter notebook but lets you call it from the command line. The program takes in the file containing the colors and the output image you want the plot to be saved to. Here's an example command line:
  `python plot_canadian_subdivisions.py random-subdivision-values.tsv test-plot.png`
- `random-subdivision-values.tsv` is an example of an output that your program should produce. The file contains two columns: (1) CSDUID -- which is the unique identifier for a Canadian subregion and (2) color_value -- which is a scalar value reflecting how much the subregion uses that word. Note that color_value can be negative and most likely will be the Getis-Ord G* value for that subdivision or a value derived from the PCA output.
- `subdivision_to_neighbors.tsv` is a tsv file with two columns: (1) the CSDUID for a particular region and (2) a space-separated list of CSDUID that should be considered neighbors in the weight matrix. In essence, you'll be creating a binary weight matrix,

where each subregion is a row and there is a 1 for every subregion in its second column, and a 0 for all other subregions. There are only 4240 subregions, so this can be a dense matrix. You'll want to construct this as a *weighting matrix* **W** for computing the spatial autocorrelation (Local Getis-Ord G*).

What To Do and Grading

This homework has three gradable components:

1. (25%) Compute the Local Getis-Ord G* for several words in your dataset and plot maps showing the Z-score values of each word for each region.[2] Include these maps in your report.
    a. Your maps should include at least one map for word that is used much more in one region
2. (50%) Compute the Local Getis-Ord G* for *all* words in the data and construct a matrix where rows are census regions and columns are words. The cells should contain the Z-score for the Local Getis-Ord G* value for the column's word in the row's census region. Then compute Principal Component Analysis (PCA) on the matrix.
    a. For the first three principal components (the largest ones), construct maps that show each regions relative values and include these in your report.
3. (25%) Analyze the principal components and examine which words are used more or less frequently in these regions. Describe what trends you see and discuss what kind of regional dialects you find.

Implementation Notes:

There are many good spatial libraries to work with in both Python and R. We recommend using Pysal for compute the Local Getis-Ord G, which has lots of documentation (e.g., this one) and a build it function. **Remember, a key learning objective is learning how to use these kinds of libraries.** The first step will be to convert the weight matrix in its provided text format into a mathematical format suitable for use with the library. (This is mostly finding the right library call). You're still welcome to use whatever library you find most suitable.

For computing PCA, you'll want to use what's known as a thin or truncated version, where you compute only the first few principal components. In sklearn, you set this with the `n_components` argument to PCA. Otherwise, you will compute the full decomposition, which is much more time intensive.

---

[2] You can interpret the z-score here as how over- or under-represented is the word's frequency (technically, its autocorrelation) with respect to the null model, which assumes no spatial configuration. Positive values mean it's overrepresented and negative values mean it's underrepresented.

Academic Honesty Policy

Unless otherwise specified in an assignment all submitted work must be your own, original work. Any excerpts, statements, or phrases from the work of others must be clearly identified as a quotation, and a proper citation provided. Any violation of the University's policies on Academic and Professional Integrity may result in serious penalties, which might range from failing an assignment, to failing a course, to being expelled from the program. Violations of academic and professional integrity will be reported to Student Affairs. Consequences impacting assignment or course grades are determined by the faculty instructor; additional sanctions may be imposed.

Submission Procedure

Your assignment is complete when you have submitted

- uploaded your code for your system as a .zip or .tar.gz to Canvas
- uploaded a PDF copy of your written report to Canvas

All of these must be submitted by the deadline or the submission will be counted late. You have three total late days to use for this project and submissions received after that time will not be graded.

Notes and Hints

- You can adjust the map plotting code as needed to make prettier maps
- For finding regional words, try looking for words associated with a place, such as names of big cities, sports teams, or local businesses.