Subreddit Similarity

Michael Snowden

March 2017

1 Introduction

This project was inspired by Trevor Martin's article on fivethirtyeight.com.

2 Data Mining

I basically did exactly the same thing as Trevor. I considered all Reddit comments that were created between the beginning of January, 2015, and February, 2017. From those comments, I ranked all subreddits by their popularity, the number of unique authors they had. I then selected the top 2,000 of those subreddits. However, I removed the top 200 of those subreddits, which produced the reference subreddits because that's been shown to produce better results. I then found all unique subreddit, significant author pairs where an author was significant iff they had posted at least 10 comments in the subreddit. From there, I compared every subreddit to every reference subreddit to find an overlap, which was how many unique significant authors the subreddits shared, but I ignored instances of only one overlapping author because it saved me an order of magnitude of storage space. This gave me a table with three columns (subreddit, reference, overlap). It's worth noting that this table doesn't include every subreddit because there are some subreddits that don't have an overlap of at least two with any of the reference subreddits. All in all, there were 40,875 subreddits compared to the 1,800 references.

3 Vectorizing

I exported the overlap table to a CSV file and constructed a space matrix C, with $m = 40,875$ rows and $n = 1,800$ columns, where $C_{i,j}$ is the overlap between subreddit i and reference j. From C, I computed the positive pointwise mutual information (PPMI) matrix, P, where $P_{i,j} = \max\left(0, \log_2 \frac{C_{i,j}}{(\sum_{k=1}^{m}\sum_{l=1}^{n} C_{k,l})(\sum_{k=1}^{m} C_{k,j})(\sum_{l=1}^{n} C_{i,l})}\right)$. After that, I normalized the rows of P. So now the i-th row of P represents the vector for the i-th subreddit.

4 Subreddit Maps

The method laid out in the FiveThirtyEight article goes like this: For each inner subreddit, x, dot it with each outer subreddit, $A_{*,i}$ this produces a vector of similarities between the inner subreddit and the outer subreddits, $s = Ax$. Divide this vector by its sum to get a vector of coefficients: $c \leftarrow \frac{s_i}{\sum_{i=1}^{n} s}$ where n is the number of outer subreddits. Use c as coefficients to find a weighted average of the polygon vertices, $p = \sum_{i=1}^{n} c_i v_i$, where v_i is the i-th polygon vertex.

This is a good notion of where subreddits lie between each other, but there is another way to do it. Let the goal be to maximize the similarity of the derived vector z to y, which is given by $\frac{z}{||z||} \cdot y$. However, if we just optimize this, we won't necessarily end up with a solution that fits into the regular n-gon because x may have negative elements, or it may not sum to 1. In order to fix this, add constraints. This produces the following constrained, nonlinear optimization problem:

$$\begin{aligned} \text{Maximize } & \frac{1}{||Ax||}(Ax)^\top y \\ \text{Subject to } & \sum_{i=1}^{n} x_i = 1 \\ & x_i \geq 0 \, \forall i \in \{1 \ldots n\} \end{aligned}$$

First, observe that

$$\frac{1}{||A\alpha x||}(A\alpha x)^\top y = \max_x \frac{1}{||Ax||}(Ax)^\top y$$

Meaning that, for any scalar α that we multiply x by, the objective function is the same. Incidentally, dividing by $\sum_{i=1}^{n} x_i$ does not change the objective function, so we can remove that constraint and just apply it at the end. Therefore, our new objective function is:

$$\begin{aligned} \text{Maximize } & \frac{1}{||Ax||}(Ax)^\top y \\ \text{Subject to } & x_i \geq 0 \, \forall i \in \{1 \ldots n\} \end{aligned}$$

Observe:

$$\begin{aligned} \frac{1}{||Ax||}(Ax)^\top y &= ||y|| \cos \theta_x \\ &\leq ||y|| \end{aligned}$$

This places an upper bound on the optimal solution.

Let $Ax = y + e$, then

$$\frac{1}{||y+e||}(y+e)^\top y = \frac{||y||^2 + e^\top y}{\sqrt{||y||^2 + y^\top e + ||e||^2}}$$

Which goes to $||y||$ as e approaches 0. So, finding a non-negative x which minimizes $||e||$ should work pretty well. So I use scipy's implementation of non-negative least squares. This isn't optimal by any means, but it tends to produce better accuracy than the other method.

So, our algorithm is this:

1. Find the non-negative least squares solution of $Ax = y$, x
2. Divide it by its sum $x \leftarrow \frac{x}{\sum_{i=1}^{n} x_i}$
3. Create the point to be used in the graph $p = Vx$ where V is a matrix whose columns are the vertices of the regular n-gon in 2-space corresponding to the outer subreddits.
4. Find the similarity $\frac{1}{||z||} z^\top y$ and display it to the user