Google Search Election Predictions

By Tony Goss

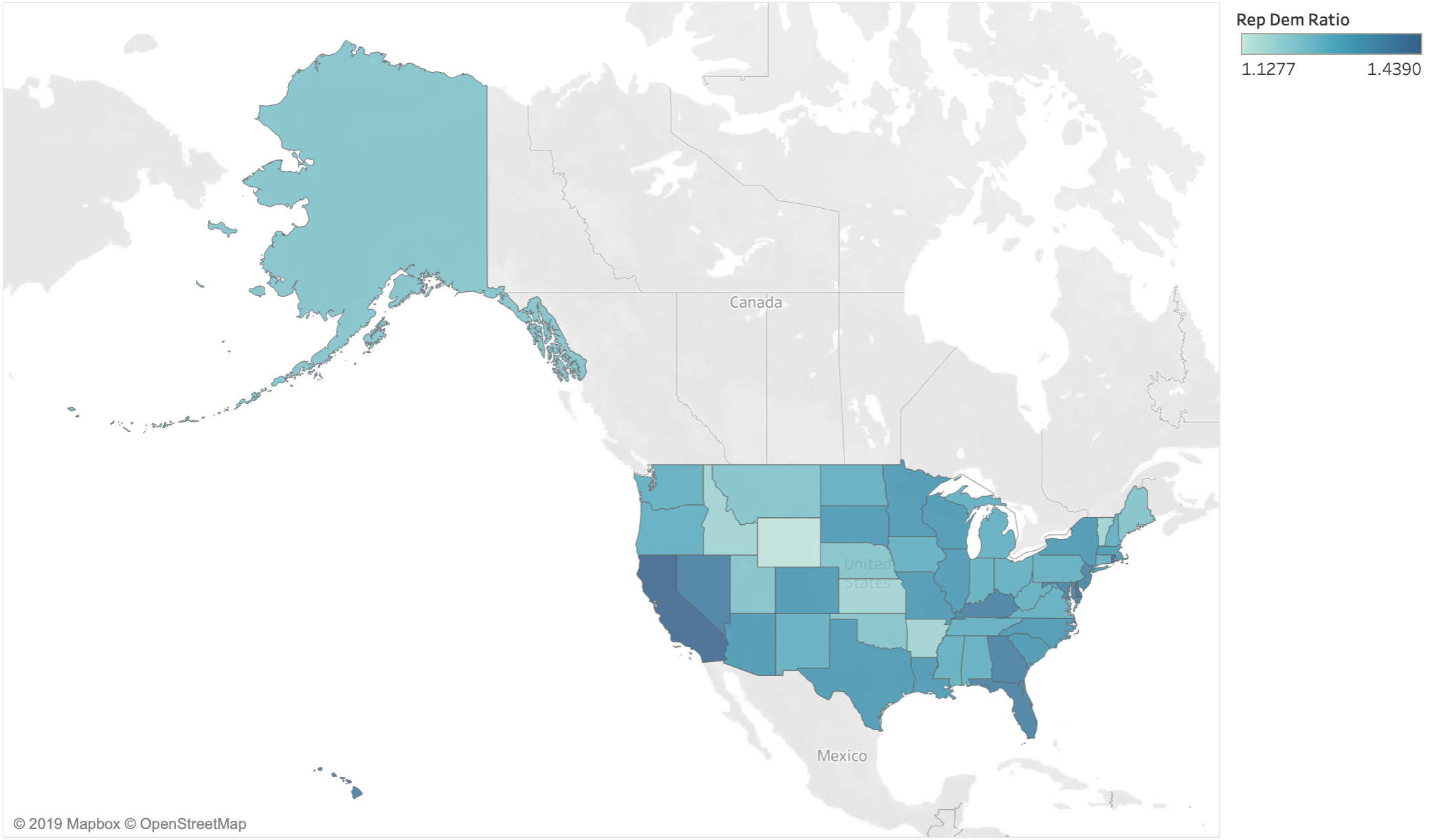
Intro

- This project extends a study done by Laura Granka, the UX Director at Google Search and Assistant.
- Laura uses Google Trends data to build a model for predicting US general election results
- Here is that paper: https://www.uvm.edu/~dguber/POLS234/articles/granka.pdf
- This paper was written in 2013, so I applied Laura's methodology for the 2016 election cycle

Data

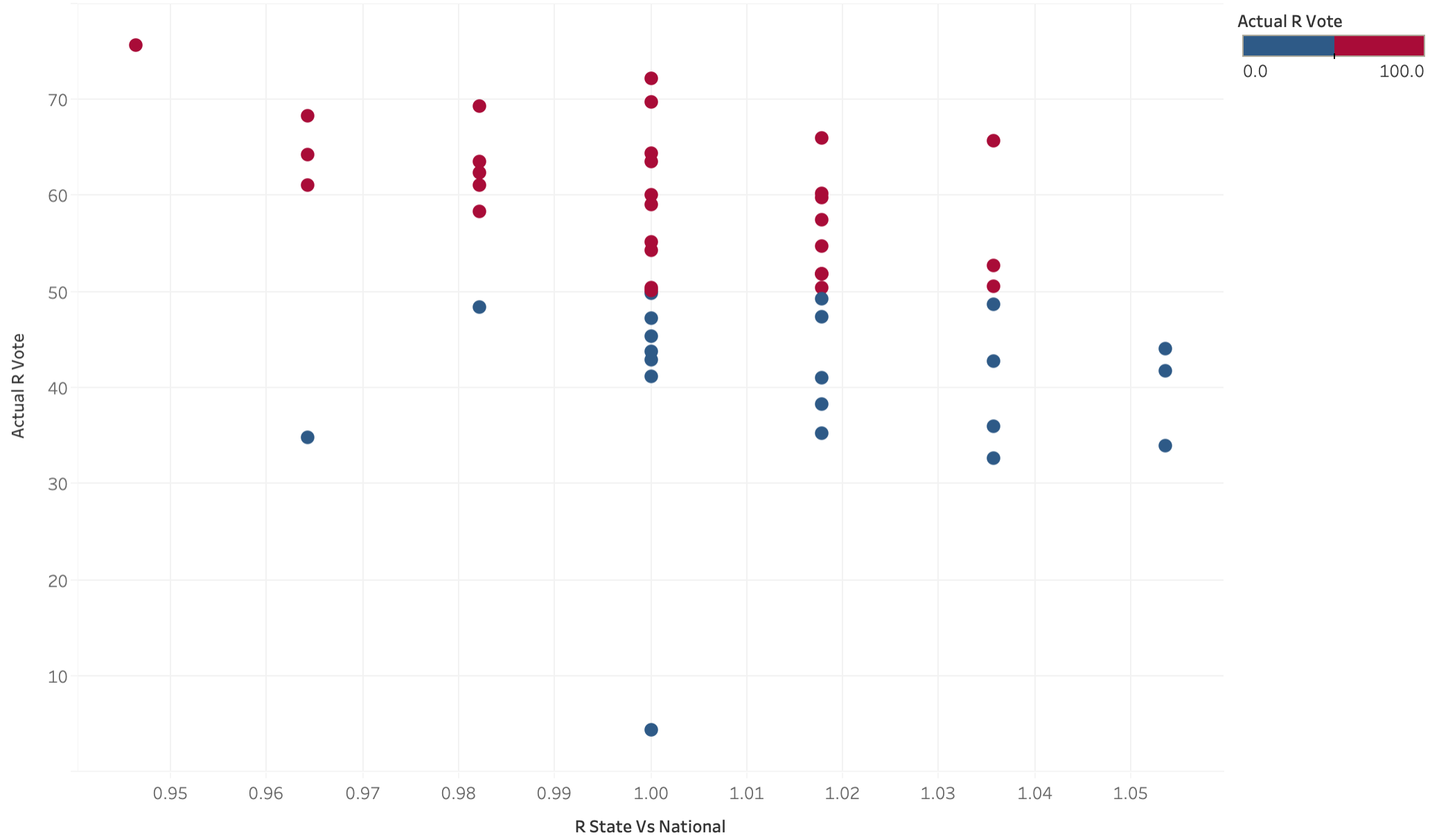
- In order to do this work, I needed Google Search data
- I used a Google Trends Javascript API made by Pat310 to gather the data.
  - For the time period of July 1 - October 31 for the elections of 2004, 2008, 2012, and 2016, I gathered state-level Trends data comparing the candidate's names
  - For example, in 2016, 56 percent of searches for either Clinton or Trump were for Trump
- Following the paper's example, I then added supplementary metrics:
  - Lag vote - The % Republican vote of the state in the previous general election
  - State/National Ratio – This metric compares the state-level search percentage for each candidate to the national percentage search percentage for each candidate
    - For example, in Delaware in 2016, the ratio of Trump to Clinton searches was 3 % higher than the national Trump to Clinton average in 2016
  - Rep/Dem ratio – The ratio of Republican to Democrat searches in each state

Ratio of Rep Candidate to Dem Candidate Google Search 2016



Map based on Longitude (generated) and Latitude (generated). Color shows sum of Rep Dem Ratio. Details are shown for State1 and Year Year. The data is filtered on Year Year, which keeps 2016.

Ratio of State Candidate Search % to National Candidate Search % 2016

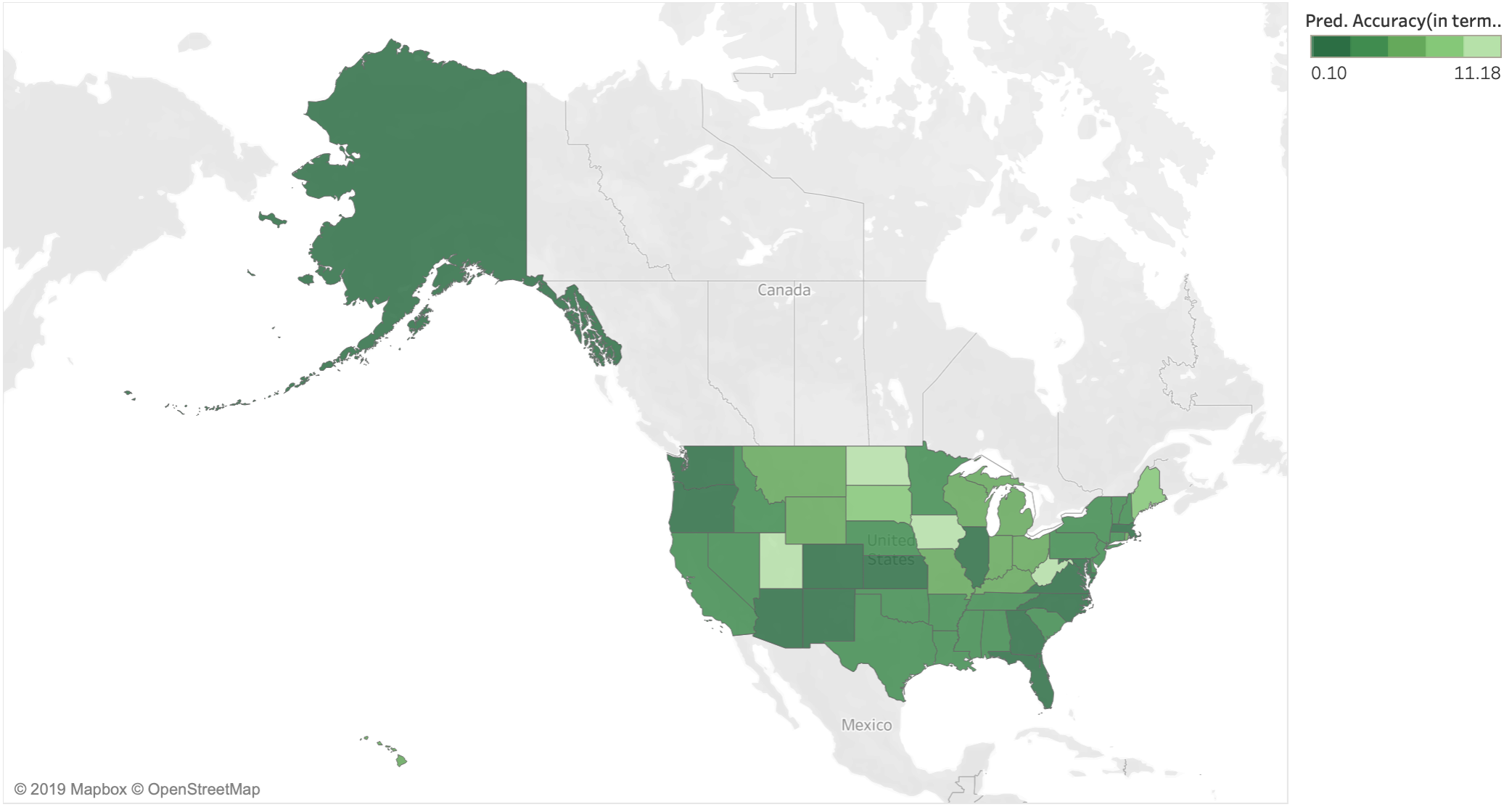


R State Vs National vs. Actual R Vote. Color shows sum of Actual R Vote. Details are shown for State1. The data is filtered on Year Year, which keeps 2016.

Model

- I ran a simple linear regression in R:
  - Model: VOTE R = LAGVOTE R + REP [St/Nat] + Rep/Dem Ratio
  - I trained 3 models: One with 2016 as heldout data and just lagvote as the indendent variable, one with a random sample of states from the four elections, and one with 2016 as heldout data and the entire model used.
- Results:
  - Baseline model (Just lagvote as predictor, 2016 is heldout data): Residual standard error: 4.294 on 151 degrees of freedom Multiple R-squared: 0.8587**, Adjusted R-squared: 0.8578**
  - Model 1 (80% of data is train set, 20% is test): Residual standard error: 3.891 on 159 degrees of freedom Multiple R-squared: 0.8993, **Adjusted R-squared: 0.8974**
  - Model 2 (2016 is heldout data): Residual standard error: 3.999 on 149 degrees of freedom Multiple R-squared: 0.8791, **Adjusted R-squared: 0.8766**

2016 Predictions from Linear Regression Model



Map based on Longitude (generated) and Latitude (generated). Color shows sum of abs_resid. Details are shown for State1. The data is filtered on Year Year, which keeps 2016.

Challenges

- Technical challenge: The Google Trends API would only allow a small number of requests at a time.
  - Solution: In order to slow down my AJAX calls, I used the setTimeout() function.
- Statistics challenge: Finding what Search data is useful:
  - I followed the paper's model, but I still have many more avenues to explore!

Next Steps

- I have improved on the preliminary lag-vote model but in order to do better, here is some additional hypotheses I would like to add to the regression:
- Order of candidate names signals support
- Searches for the n-word - from Seth Stevens-Dadowitz's book Everybody Lies
- College education in the state
- Interaction terms between features