Using Google Street View to Analyze Voter Engagement

Daniela Shuman, Elie Eshoa, Luke Stoner

Data Summary

Outcome Variable

Our outcome variable is the binary variable of the outcome of the 2021 election. This variable was calculated by taking the 2021 midterm election results from Bloomberg Election Results and the winner of the election, or the incumbent, determined the party that won the voting district. This voting district is labeled as the [state ID] - [district ID].

Independent Variable (The Image Dataset)

For our final X dataset, we intend to have 10,000 randomized street view images from across the United States. We would use an 80/20 train-test split, using 8,000 images for training and 2,000 for testing. Based on a trial of our web scraping process, downloading 100 images took approximately 8 minutes (which can be seen under the directory **`trial_data`**). Therefore, we would expect the total collection of our dataset to take just over 13 hours. We plan on spreading out our collection to five collections of 2,000 images, each collection taking roughly 3 hours. Details about the collection process are explained later.

Regarding the size of the dataset, each image will have a final resolution of 250x170x3, resulting in a file size of approximately 10 kilobytes. Therefore, our final dataset would only be 100 megabytes, a very manageable size that can be uploaded to JupyterHub. Ultimately, while having more images at a higher resolution would be ideal, based on the constraints of our processing power and data collection we believe our plan is suitable for this project.

Insights & Noteworthy Findings from our Variables

Outcome Variable

Both the 2018 and 2020 variations of the congressional show strong balance across Republican and Democratic districts. The house was very competitive across this period of time.

Independent Variable

Based on multiple trials of the data collection process, we believe our final image dataset will contain the necessary images to answer our research questions. Images ranged from city

streets to suburban homes, desert roads, and corn fields. That said, one potential concern is that the dataset is slightly biased toward rural areas. Basically, there are just more roads in rural areas than in urban areas. However, we do believe we will ultimately have enough variation since our randomized data scraping spans the entirety of the US.

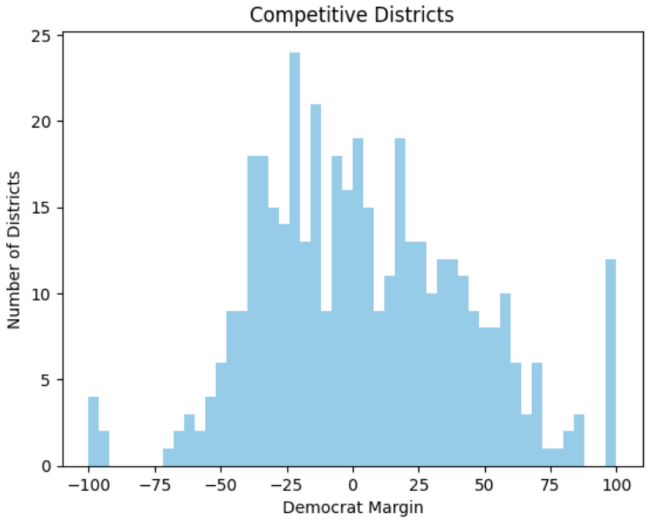
Feature Extraction from the Independent Variable

Based on our data insights, one potential challenge with using street view images as the independent variable is that it can be difficult to extract useful features from raw images that can be used in predictive models. One approach is to use image segmentation, which involves dividing the image into different regions based on common characteristics such as color, texture, and shape. This can be useful in identifying important objects or features in the image, such as buildings, roads, or vehicles, which could potentially be related to voter engagement. These features can be used as predictors in our model.

Regarding feature extraction methods, using computer vision models like U-Net transfer learning (209b A-Sec 4) can both save time and improve accuracy. Saving time can be accomplished by loading pre-trained weights, for instance, and improving the accuracy can be achieved by fetching fine-tuned weights for our particular tasks, which may include spotting particular objects such as trees, roads, flags, as mentioned above. Refer to the implementation section for more possible approaches to achieving our tasks.

Visualizations

The below visualization outlines the competitiveness landscape of the 2021 elections by observing the difference in the percent of votes the Democratic candidate got vs. the Republican candidate got in the general election for the representative. One way to improve the accuracy of the model would be to treat competitive districts differently than non-competitive districts. The code for this visualization is in **`Data Processing.ipynb`**.



It is difficult to visualize our X data systematically because we cannot extract particular features out of our images without running a computer vision algorithm on it. This had been discussed in the feature extraction

Data Collection Process

Outcome Variable

We used Bloomberg's Election (source) results to classify each election district as Democrat or Republican. From this source, we received the winning elector party as well as the margin between Democrat and Republican. Each voting district was denoted by a voting district ID. We combined this with GEOID, which classifies each district as a census tract.

Independent Variable

Initially, we planned on using the following dataset (source), which contained Google Street View images from Pittsburgh, Orlando, and Manhattan (total of 60k images). However, as we noted, this would impose significant challenges, mainly due to the fact that these are all metro areas that lean majority Democrat. Therefore, we would not have much variation in our X variable.

However, we now have a new plan to create our own dataset of 10,000 street view images which are taken from randomized locations across the United States. We are able to use web scraping from the following site (source) to gather images and associated zip codes. We then appropriately crop and resize the images and match zip codes to congressional districts via the following site (source) to create our final dataset. The code that completes each of these processes is found in **`Milistone3.ipynb`**. Examples of the images are found below:

Initial Scraped Screenshot

Final Cropped/Resized Image

Project Question

Can we use Google Street View images to analyze and predict voter engagement in the 2021 midterm elections, and how does this vary across different geographic regions and demographic groups? We will be guided by the tasks we wish to solve based on our images (the independent variable). Refer to the feature extraction section that outlines possible features to be used to answer our question of predicting voter engagement.

Baseline Model & Planned Implementation

We outlined a baseline CNN model in our notebook. We plan on using an inception model (GoogleNet) for our baseline, largely based on the model from the lab, slightly modifying it to fit our problem. After fitting a baseline model and accessing its accuracy, we can move forward with feature extraction and segmenting our images to identify particular features. We can use U-Net transfer learning or other models such as ResNet, RNNs, or even transformers.

We will use JupyterHub for the implementation of our neural networks and anticipate that each model will take roughly 30 to 60 minutes to run with the full dataset. Therefore, we will likely run trials with subsets of the data (ex. 1000 images) to test accuracy.