Report

Overview

The goal of this project was to assign probabilities to the upcoming Georgia Senate Election Runoff between David Perdue and Jon Ossoff. I approached this project with a combination of prior political science understanding as well as a developing knowledge of simulation techniques. The end result of this project is a graphic that shows the user the projected probabilities for each candidate. Additionally, this simulation can be updated as new polls come in.

Implantation

Every file within this repo is self-contained and will run independently. Within the repo is a package called `georgia` which holds a few of the key functions used in the project. The three functions it contains are `probability_winning_plot`, `probability_dataset`, and `initial_poll_cleaning`. The goal of these three functions is to take dataset from `fivethirtyeight.com` and allow it to be easily implemented in my probability forecast model. `initial_poll_cleaning` takes in a downloaded `.cvs` file that is saved as a variable and does the data manipulation necessary to proceed with the forecasting. The function `probability_dataset` works on this by doing some of the more complicated data manipulation, involving logical statements, and makes it ready for the last function, `probability_winning_plot`. `probability_winning_plot` then plots the probabilities in a way that allows for dynamic poll values.

Bootstrapping and Mapping

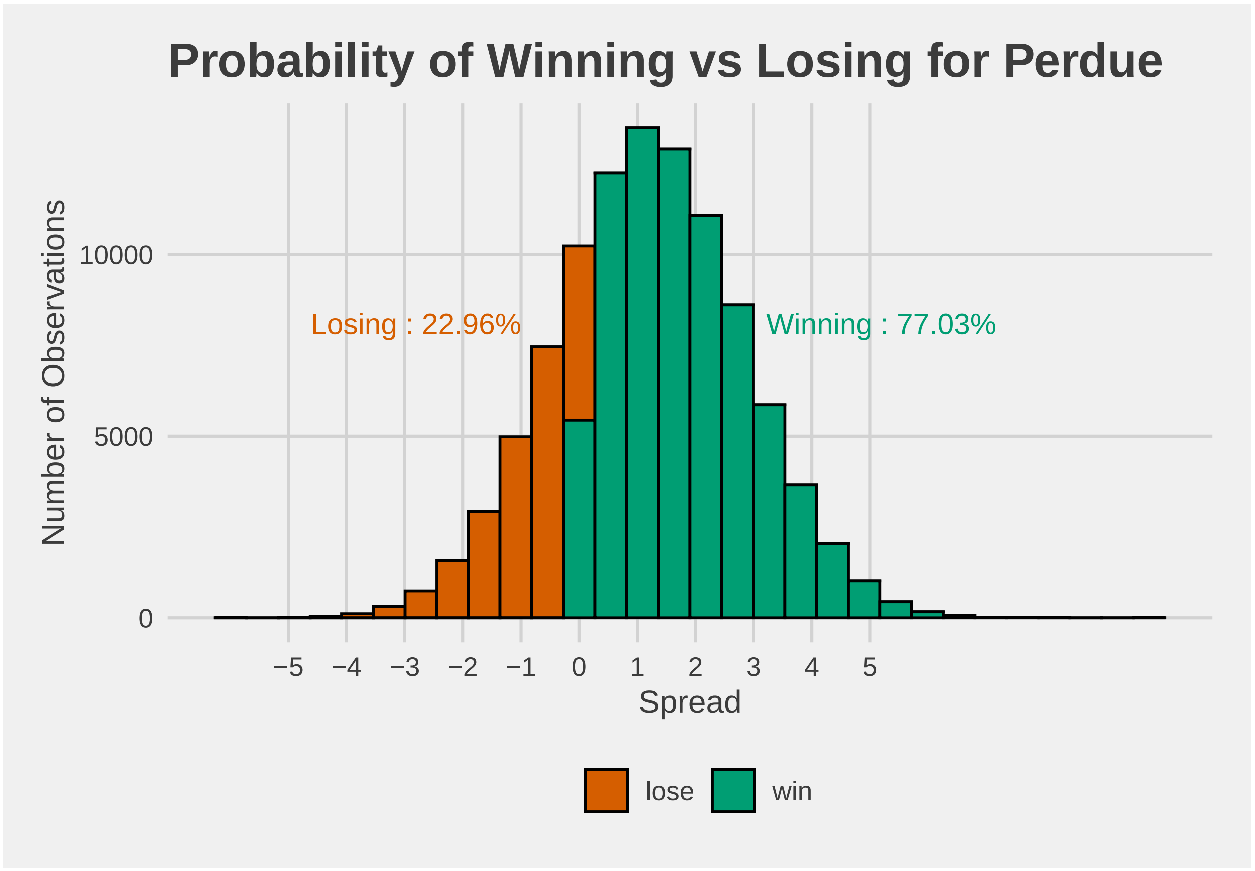
In addition to the functions described above, here are the actual simulation techniques used to come up the probabilities. At the time of writing there are 114 polls for this particular Senate race. Although this is actually a decent amount of polling compared to most senate races, in my opinion it was not enough to make an accurate prediction. My first attempt to model the uncertainty in the election can be seen in my file `01_Johnson_Disribution.Rmd`. Although this did not pass the mathematical muster required of an election simulation it was an important part of the process towards finding a good way to go about the project. It led to the implementation of bootstrapping, which can be found in `02_Bootstrap.Rmd`. After cleaning the polling dataset, resulting in the named dataset `boot_data`, I preformed the following bootstrap:

```
map(1:10000, ~sample(boot_data$actual_spread, size = length(boot_data), replace = TRUE))
%>% map_dbl(mean)
```

This line of code takes samples from the `boot_data` with replacement and then maps over them ten thousand times and lastly extracts the mean. This allows my data to achieve normality and make some predictions.

Assigning Probability Values

The final step was to take these bootstrapped spreads and overlay them on a normal distribution using the `rnorm` function and adjust them by each candidate's current polling advantages or disadvantages. This can be found under the `Analysis` folder in file `03_candidate_probabilities.Rmd`. Below is the result if the user chooses to see the probabilities for Perdue:



In order to make sure these projections are relatively accurate in `04_fake_polling.Rmd` I make fake polling data based off of a uniform distribution. The function `fake_polling` allows the user to input the minimum and maximum values and then simulate a uniform distribution. It returns a candidate's winning vs losing probabilities given that underlining distribution. Experimenting around with these numbers allows the user to check that the simulation is working correctly. For example, if the inputted minimum and maximum are equally far from zero then the resulting forecasted probabilities should be roughly fifty-fifty, and that is indeed what happens.

The last file, `05_weighted_sampling.Rmd` addresses the likelihood of what polls are chosen to be bootstrapped. The function `boot_data_weights` allows the user to adjust the weights of each pollster rating category. And then running the data through a similar procedure as before and making use of the `probability_winning_plot` function to visualize the probabilities.

Learning Outcomes

When going into this project I initial had some pretty ambitious ideas but quickly found that they were either out of the scope of this class or I simply did not have enough time to complete. Regrettably this project does lack some of the simulation rigor I was hoping for and I do not think it demonstrates my full understanding simulation techniques. But this is not to say I have not learned a lot and done a lot of work along the way. I came into this with little to no understandings of GitHub. As a result I have gained valuable knowledge into how to organize a repository and overall how to be more professional with my work.

References

Data sourced from fivethirtyeight.com

The following are some articles that helps guide my thinking:

- https://medium.com/planscore/how-its-made-a-planscore-predictive-model-for-partisan-elections-50c14bfc6e8
- https://projects.economist.com/us-2020-forecast/president/how-this-works