Project Report for Stats 170B, Spring 2022

Project Title: Predicting Political Election Outcomes at the County Level From 1990 to 2020

Student Names
Gurpal Kundi, 27198663, kundig@uci.edu
Roy Minato, 16641795, minator@uci.edu
Shou Shimaya, 41129568, shimayas@uci.edu
GitHub: https://github.com/shoushimayaUCI/ElectionPredictions

1. Introduction and Problem Statement

Election prediction has traditionally been done in two senses. One is in an academic sense where the objective is to look at what factors influence the outcome of an election. The other is more campaign related, where a campaign might collect data and conduct polling to determine where to put resources. A modern example of this is the use of social media posts to predict outcomes. In this project, we aim to predict election outcomes at the county level for presidential, senatorial, and gubernatorial general elections in the United States from 1990 to 2020, and explore which features are especially useful for doing so. Features we have incorporated into our models include state level polling data, national economic metrics, and county demographic data. Since the overwhelming majority of and interest in elections are between the democratic and republican parties, we will not be including elections in which democrats or republicans did not nominate a candidate, or elections between a third party candidate and a democrat or republican candidate.

The motivation behind our project is to essentially understand how different geographies in the United States, counties in our case, politically identify and if their political preference in a given election year can be reliably predicted by observing features that describe the county population, their past voting trends, and broad national economic metrics. We think what makes our project especially interesting is that most of the literature regarding election forecasting is almost exclusively done at the national or state level with little to no consideration for counties. We believe that these current methodologies could lead to underrepresentation of populations and, as a result, less accurate predictions such as those conveyed by unrepresentative state level polling for the 2016 presidential election in several states.

2. Related Work:

One primary source we used to help us with our approach is the popular political forecasting site Five Thirty Eight. Five Thirty Eight performs sophisticated election forecasting using combinations of population predictors along with polling data. They use dynamic methodologies that change throughout an election cycle, where different predictors are given varying degrees of importance as the election cycle proceeds. They also create different models that include and exclude polling data to compare performance.

In terms of the types of modeling we intend to perform we had to incorporate that our data is structured as a time series with multiple predictors, with the outcomes for elections for each county being the response variable. Traditionally, time series modeling is done using auto-gressive modeling such as ARIMA (Autoregressive Integrated Moving Average). However, this is often limited in the number of predictors that are feasible to compute. A modern approach is to use Recurrent Neural Networks such as Long Short Term Memory networks. This allows for a larger number of parameters to be included, and allows flexibility in the response variable. Helpful resources were the textbooks *Forecasting: Principles and Practice (3rd ed)* by Rob J Hyndman and George Athanasopoulos, and *Deep Learning for Time Series Forecasting(1.6 ed)* by Jason Brownlee.

3. Data Sets:

Election Data:

For our election data at the county level, we used CQ Press Library's Voting and Elections Collection accessed through UCI Library (https://library.cqpress.com/elections/download-data.php). These data sets include our response variable which is the party that won the election in the county (democrat 'D' or republican 'R'). We use this response information and the percentage of votes received by both parties in each election to create new features such as the three previous election outcomes and vote percentages for each party for the same election type in the county.

State Polling Data:

Since election polling is overwhelmingly done at the state level, we resorted to using state level polls for each county in that state for the matching election. For presidential elections, we used polling averages calculated by FiveThirtyEight (https://github.com/fivethirtyeight/data/tree/master/polls) while we used Cornell University's Roper Center for Public Opinion Research for senatorial and gubernatorial election polls (https://ropercenter.cornell.edu/exit-polls/state-election-day-exit-polls).

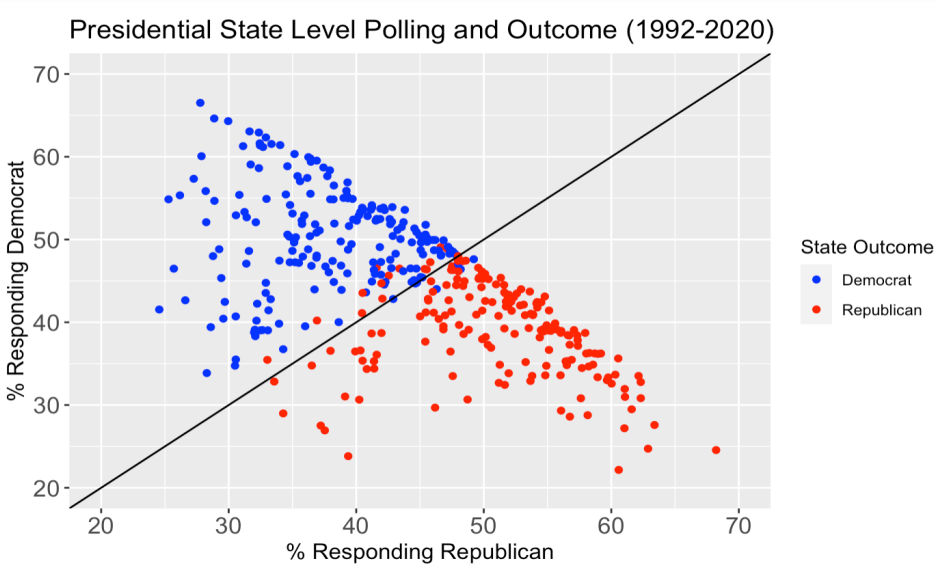

Figure 1

The visual in figure 1 above conveys how accurate FiveThirtyEight's weighted averages of polls are in predicting election outcomes at the state level. If a point is over the added line, democrats polled higher than republicans so we would expect a democrat win. In contrast, we expect a republican win when the point is under the line.

National Data:

We decided to include broad economic information about the entire United States and joined them to each county based on year. To do this, we used the Bureau of Labor Statistics to obtain the national unemployment rate each year (https://www.bls.gov/cps/tables.htm#empstat) and we web scraped a table from MacroTrends to gain information about real GDP growth and its annual change each year in the United States (https://www.macrotrends.net/countries/USA/united-states/gdp-growth-rate).

County Demographic Data:

We obtained race and sex combination population counts in each county from the Census Bureau (www.census.gov). There were three files in total, one for each decade from 1990 to 2020. These files differed slightly from one another as later files included more detailed information about individual races that were previously grouped together in previous files. In each file, a row refers to a county while columns refer to the number of people in that county that fit the race and sex combination specified in the column header. The Census Bureau also displays counts of people in each race and sex combination group who identify as ethnically Hispanic which we further integrate as part of our features. We converted the data from counts into proportions.

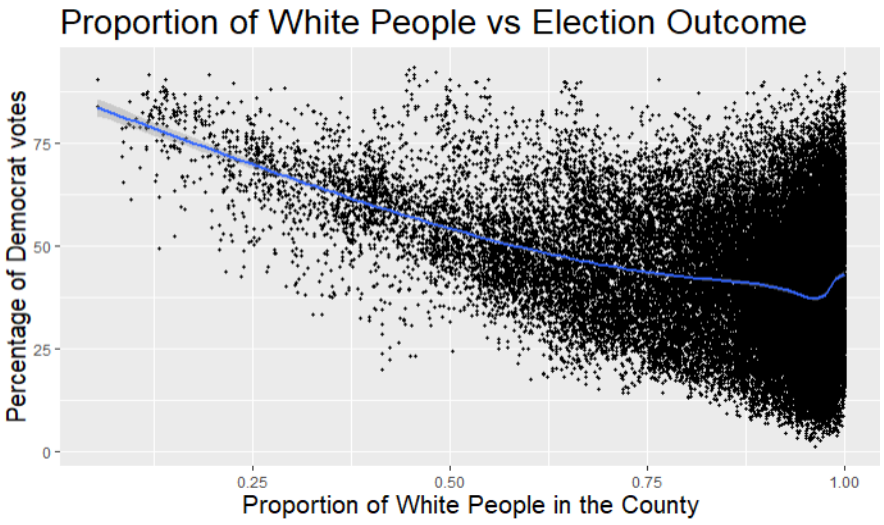

Figure 2

We can see how, in the case of the above plot, race distributions in a county can offer some insight into election outcomes at the county level. When the proportion of the White population in a county is less than fifty percent, a sizable majority of county outcomes are democrat as the points lay above fifty percent of democrat votes compared to republicans. This also conveys that counties with high proportions of racial minorities might more often choose to vote democrat instead of republican.

County economic data was added in the form of the annual unemployment rate for each county. This data was obtained from the Bureau of Labor Statistics (https://www.bls.gov/lau/). Each year from 1990 to 2020 has its own file, but because the format and structure of their recorded data does not change over time, we simply concatenated them into a single table using Pandas. We also use the column detailing labor force size in each county as a measurement of the level of urbanization in a county.

We obtained proportions of the highest level of educational attainment in each county among adults from the United States Department of Agriculture's Economic Research Survey (https://www.ers.usda.gov/data-products/county-level-data-sets/download-data/). However, since this file only includes decennial values for each county, we linearly interpolated values as we thought it would be a sensible method of obtaining values that although are not exact, would be reasonable estimates. The categories include: less than high school, high school only, some college, and bachelors or higher.

Lastly, we obtained population counts of five-year age groups from the Census Bureau (www.census.gov). We had to download many files because of how the Census Bureau chose to publish the data. For the 1990's, we downloaded nine separate files where each file includes data for all counties in a given year. For the 2000's data, we instead had to download fifty files where each file contained data on the years 2000-2009 for each county in a specific state. 2010-2020 data was instead published just as a single file. We converted the counts in the original datasets into proportions. As we handled the Census Bureau's data for race and sex combination populations, we repeated this process of pre-processing

each decade's data separately before getting them to similar formats where we could simply concatenate all the data into a single table.

Combined Dataset:

Once we were able to pre-process all the data from a source into its individual table, we began the merging process. To merge state polling data to our election data, we utilized election office, state, and year columns to ensure polls for a state would be joined to each county in that state for that particular year and particular election. Since we wanted national level data to be shown with each county, we simply joined by the year column so relevant national economic measures would be joined. To join our county level demographic data, we had to join on year, state, and county columns. Using year and county ensure a county is joined with data relevant to the election and we include a join on state to account for the fact that several counties in different states can have the same name.

In summary, our features are composed of the three previous elections outcomes in the county, statewide polling data, party incumbency status, national unemployment rate, annual national GDP growth and year over year change, race and sex combination populations by county, county educational attainment rates, populations of five year age groups by county, and county unemployment rate.

	office	state	year	county	repcandidate	repstatus	demcandidate	demstatus	winningParty	LastPartyWon	...	age5559	age6064	age6569
0	President	Kentucky	2020	Bourbon	Trump, Donald J.	Incumbent	Biden, Joseph R. Jr.	Challenger	R	R	...	0.073363	0.068288	0.060097
1	Senate	Kentucky	2020	Bourbon	McConnell, Mitch	Incumbent	McGrath, Amy	Challenger	R	R	...	0.073363	0.068288	0.060097
2	President	Georgia	2020	Camden	Trump, Donald J.	Incumbent	Biden, Joseph R. Jr.	Challenger	R	R	...	0.060230	0.058298	0.048801
3	Senate	Iowa	2020	Shelby	Ernst, Joni	Incumbent	Greenfield, Theresa	Challenger	R	R	...	0.079615	0.079003	0.066492
4	President	Iowa	2020	Shelby	Trump, Donald J.	Incumbent	Biden, Joseph R. Jr.	Challenger	R	R	...	0.079615	0.079003	0.066492
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
74495	Governor	Wisconsin	1990	Calumet	Thompson, Tommy G.	Incumbent	Loftus, Thomas	Challenger	R	R	...	0.039304	0.038751	0.032846
74496	Governor	Wisconsin	1990	Burnett	Thompson, Tommy G.	Incumbent	Loftus, Thomas	Challenger	D	D	...	0.052102	0.067108	0.064747
74497	Governor	Wisconsin	1990	Buffalo	Thompson, Tommy G.	Incumbent	Loftus, Thomas	Challenger	R	R	...	0.046467	0.049417	0.046983

Figure 3. Sample Records

The dimensions of our final combined dataset are 74500 by 83. All the raw data from the sources above, their wrangled counterparts, and the final combined dataset are all stored locally and in a shared Google Drive folder.

4. Overall Technical Approach

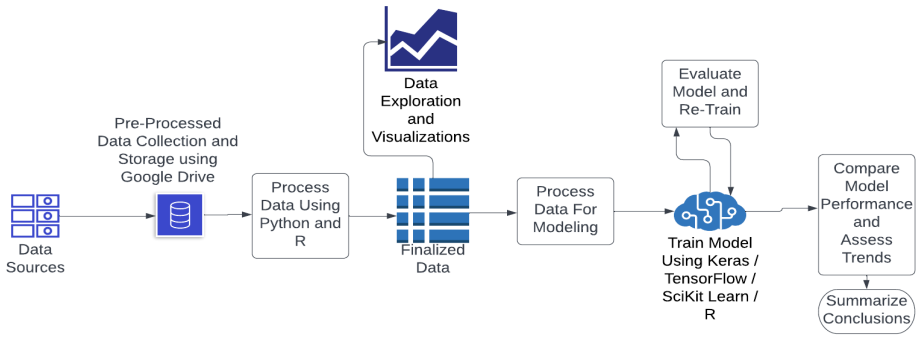

Figure 4. Pipeline

A picture of our overall technical approach is shown above in Figure 4. Every aspect of our project is organized in Google Drive, including the raw and wrangled data, Python code, and R code.

Data Management and Preprocessing:

We wrangle and preprocessed out data from our data sources as outlined in the data section above. We organized our preprocessing code such that we have a Jupyter Notebook for each data source which wrangles and creates a single processed table, and a separate notebook to merge all of them together. Python libraries used include Pandas, NumPy, and glob to read multiple files from the same directory.

Data Analysis and Machine Learning:

Once our wrangled datasets were merged into one combined dataset, we began preliminary data analysis by visualizing the relationships between several features and our response of the winning party in the county using R. This analysis offered us some insight into how our features could possibly be indicative of the county outcome. Examples of this are in Figures 1 and 2 above.

Given our project goal amounts to a binary classification problem, all modeling was directed towards this type of outcome. Before any modeling it was necessary to convert all categorical variables into numerical format, and we also performed scaling transformation to the [0, 1] interval due to our features having different scales. We then removed obvious dependent features, such as democrat and republican vote percent scores, where only one is needed since they both sum to 100.

We will train and compare 5 different machine learning models.

Machine Learning Models:

- Logistic Regression: The basic idea behind single perceptron logistic regression is to model the response variable as a probability. Statistically this corresponds to the response being modeled as a binomial random variable with the probability parameter being a linear combination of the feature variables input into a sigmoid function. Our data is a time series and so violates the assumption that all observations are independent. To deal with this we will include in as our features previous election results for a given county. We will perform this model in both Python using the LogisticRegression class and in R using the glm model. Model assessment will be done using accuracy scores in Python. In R we will observe model coefficients and perform statistical tests using p-values to assess significance among these coefficients.
- Neural Network: Once we have our baseline logistic regression model we will attempt to use other more complex classifiers. First we tried to implement a neural network multi-layered logistic regression classifier. This type of classifier uses a feed-forward neural network with hidden layers and a binary output perceptron. Advantages of this type of model is that they can capture non-linearities in the features. Some disadvantages of using a neural network are that random weight initializations can lead to different validation accuracy, they require tuning hyperparameters, and they are sensitive to feature scaling. We will try to create a network with enough hidden layers and nodes per layer so that sufficient complexity is captured, but not too large so as to overfit, as well as require a long time for the model to converge.
- Random Forest Classifier: It makes multiple decision trees that use a subset of all the features we have. Each tree is trained, and when making predictions, random forest classifiers ensemble these trees by taking the most popular result. By making multiple uncorrelated submodels and ensembling, it accounts for variability in the data, which reduces overfitting, bias, and overall variance. Thus, it beats the performance of one decision tree with all the features.
- Gradient boost / AdaBoost: In boosting methods, it makes multiple models sequentially in order to reduce the bias. Each model will be trained based on the previous model, trying to correct the mistakes the previous model made. Finally, the algorithm chooses the most popular prediction of all the submodels

5. Software

- Jupyter Notebooks with Python (Pandas, NumPy, Matplotlib, Keras, sklearn, glob, BeautifulSoup) for data processing and model learning and assessment. Data processing included cleaning scripts, reading many files of the same format in a directory into one dataframe, as well as web scraping a simple table for information of national GDP over time.
    - Seven notebooks were used for data cleaning and preprocessing, one for each of our broad general features: County age group populations, educational attainment rates, election data, race and sex combination populations, county unemployment rates, national unemployment rates, and national GDP growth and year over year change. The input for all notebooks were the raw data files from our sources.
    - One notebook was used to merge the cleaned outputs of the seven aforementioned notebooks and produced our combined dataset.
    - One notebook was used for cleaning data to be used for machine learning.
    - Two notebooks were used for data visualizations.
    - One notebook was used for model preparation, training, and assessing models using sklearn.
- One R file was used to wrangle and preprocess the Roper Center's senatorial and gubernatorial survey polls. We chose to use R in this case because the files were available in SPSS format which we figured out how to read and use in R, but could not find a corresponding Python library. One R file was used for performing logistic regression modeling.
    - The R file for wrangling and data processing uses the following packages: foreign, haven, and dplyr. The foreign and haven packages were both used for their ability to read SPSS files. We used foreign's read.spss() function for most files however ran into issues when we tried to use it for older SPSS files for which we used haven's read_por() function. Finally, the dplyr package was used to add year columns to the poll data using the mutate() function and the select() function to discard non-relevant information.
    - The R file for logistic regression uses the glm package for modeling.
- We used R and Python's matplotlib / seaborn library packages to create visualizations in the data analysis portion of our project using the ggplot2 package.

6. Experiments and Evaluation

The following figures represent training and testing accuracy evaluations of our baseline logistic regression model as well as our determined best model of the four non-linear models we attempted: multiple perceptron logistic regression, random forests, adaboosting, and gradient boosting. We ran models on each of the types of elections separately, and one for all election types combined. When we ran our models we structured our training and testing sets for each year of elections. For each election year being forecast, training was performed on all relevant election results from previous elections only. Then testing was done on that year's election data. Obviously, this meant that as the year increased, so would the size of the training set, which is why we see the general trend of better results for later elections.

When determining our best model, we looked at several aspects of performance. We looked at how the model generalized to all types of elections. We also looked at how consistent the model performed as the year increased. We expected performance to increase with year as explained above, but with some models, such as with logistic regression, there was quite a lot of variance and fluctuations. This showed us that the model was picking up insignificant signals in our data. The random forest classifier was the best in having a consistent trend for all election types. A summary table comparing all models for the 2020 election is in Figure 7. We used the 2020 election year as the best assessor of our models because it was the election with the most data to train on.

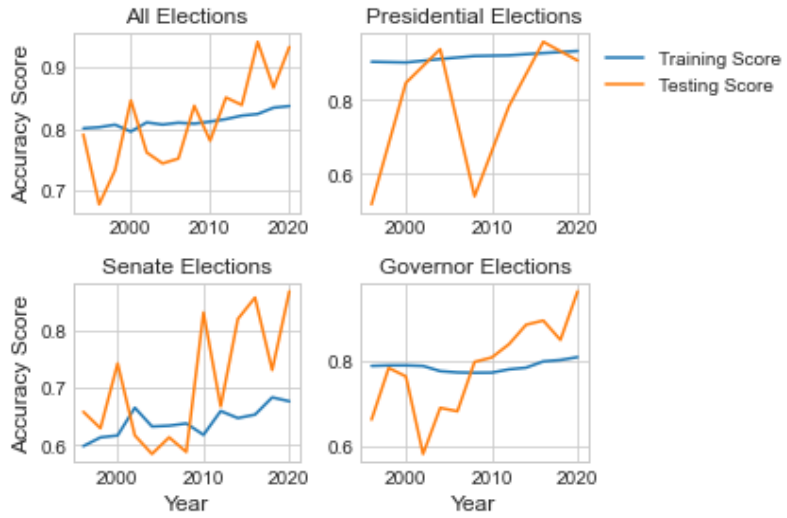

Figure 5. Logistic Regression Classifier

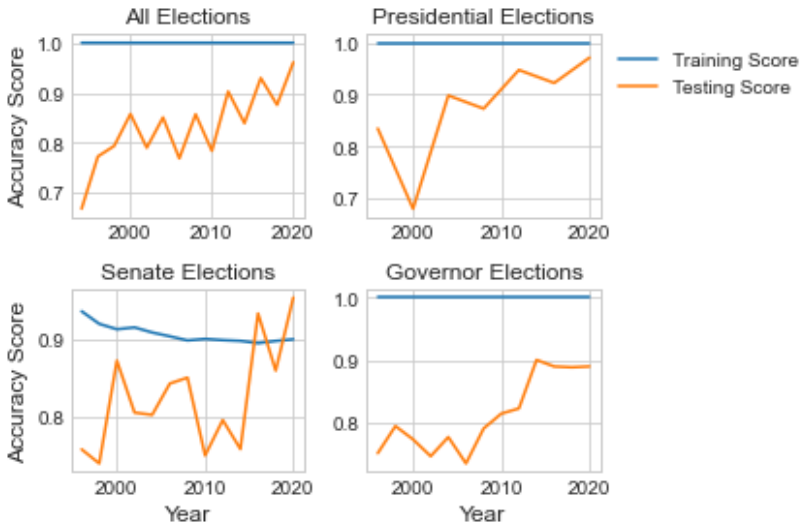

Figure 6. Random Forest Classifier

	All Groups	Presidential	Senate	Governor
Previous Result	0.942410	0.975060	0.907285	0.893578
Logistic Regression	0.933027	0.905706	0.867550	0.963303
MLP Classifier	0.864397	0.939187	0.825777	0.834862
Gradient Boosting	0.956578	0.973352	0.947020	0.875229
Ada Boosting	0.945170	0.972668	0.919002	0.873394
Random Forest	0.960810	0.971302	0.953133	0.889908

Figure 7. 2020 Election Testing Accuracy Scores by Model and Test Group

Our results were altogether promising. We had initially thought to use logistic regression as a baseline model. However, when performing our evaluation we found that the logistic regression model was not very good at predicting election outcomes. A better standard was to compare our models against using a very simple model that chose the outcome solely based upon that last election's outcome, grouping by election type and county. We can see that in each election type, different models perform better than using the previous result. The best overall was the random forest classifier, which had good training and testing accuracy. We believe that the random forest, as well as the boosting models were the best performers due to their ability to build more complex models from simpler ones. This aids in increasing the complexity of the final model without overfitting. We had thought our neural network would be able to perform better than it did. Perhaps we would be better trying more complex architectures that better represent the time series nature of our data, and that capture the relationships amongst the features while filtering out noise.

In R we tested single coefficients being equal to 0 or not using the t-test, and groups of coefficients being all equal to 0 or not using the Chi-squared test. We looked at each election type separately. Our results are summarized as follows:

<u>Statistical Tests on Logistic Regression for President Election:</u>

- Most important Features ($p < 2e\text{-}16$):
    - Democrats Vote Percentage in the previous election
    - Population gender and race distribution
    - Polls of Democrats and Republicans
    - GDP Growth
    - Annual GDP Growth
    - Population education level distribution
    - Interaction term between year since 1990 and poll results
- Significant Features ($p < 0.01$):
    - Age distribution ($p = 1.792e\text{-}12$)
    - Incumbent statuses (1.186e-06)
    - County specific unemployment rate (1.35e-06)
    - National unemployment rate (1.81e-06)
    - Democrats Vote Percentage 2 elections ago (5.94e-06)
    - Democrats Vote Percentage 3 elections ago (0.008053)

<u>Statistical Tests on Logistic Regression for Governor Election:</u>

- Most important Features ($p < 2e\text{-}16$):
    - Democrats Vote Percentage in the previous election
    - GDP Growth
    - Incumbent status of Democrats and Republicans
    - National Unemployment
    - Age distribution
    - Population gender and race distribution
    - Polls of Democrats and Republicans
- Significant Features ($p < 0.01$):
    - Democrats Vote Percentage 3 elections ago (2.73e-14)
    - Population education level distribution (9.62e-14)
    - Winning Party of the previous election (6.29e-13)
    - Year since 1990 (3.49e-10)
    - Winning Party in 3 elections before (2.31e-08)
    - County specific unemployment rate (4.57e-08)
    - Matching outcomes (3.91e-06)

- Democrats Vote Percentage 2 elections ago (2.38e-05)
- Interaction term between year since 1990 and poll results (2.668e-05)
- Annual GDP Growth (6.37e-04)

Statistical Tests on Logistic Regression for Senate Election:

- Most important Features ($p < 2e\text{-}16$):
  - Democrats Vote Percentage in the previous election
  - Democrats Vote Percentage 2 elections ago
  - County specific unemployment rate
  - GDP Growth
  - National Unemployment Rate
  - Winning Party in the previous election
  - Winning Party in 3 elections before
  - Age distribution
  - Population gender and race distribution
  - Incumbent statuses for Democrats and Republicans
  - Population education level distribution
  - Polls of Democrats and Republicans
- Significant Features ($p < 0.01$):
  - Year since 1990 (1.44e-12)
  - Interaction term between year since 1990 and poll results (8.158e-06)
  - Office Matching Outcome (6.29e-4)

As an example, this is coefficient of 5 most important covariates for presidential election model:

	Estimate	Std. Error	z value	Pr(>|z|)
LastDemVotePercent	3.725964e-01	1.240580e-02	30.0340541	3.526894e-198
GDP.Growth....	-2.093470e+00	9.379302e-02	-22.3201012	2.357555e-110
Annual.GDP.Change....	1.442390e+00	9.199284e-02	15.6793732	2.093093e-55
X._Bachelors	9.367071e-02	9.022522e-03	10.3818769	2.998220e-25
DemPoll	1.820085e-01	1.929151e-02	9.4346463	3.923155e-21

For example, an interpretation of the coefficient for presidential election that indicates the percentage of population with bachelor's degree is as follows:

- $\beta = 0.09367071$
- Given all other variables, when the percentage of population with bachelor's degree increases by 1, the expected odds of Democrats winning the election increases by a factor of $e^{\beta} = 1.098198$

In conclusion, even given previous results and polling, there are significant social, demographic, and economic factors for predicting election results. There are differences for each election type. Presidential elections had the highest accuracy, and they rely less on social, demographic, and economic factors, perhaps because the poll results were more accurate. Governor elections were the hardest to predict, perhaps because they had the least available and less reliable data. Senate elections are the "noisiest", perhaps because they occur more frequently and are subject to higher turnover.

7. Notebook Description

Seven Jupyter notebooks were used for data cleaning and preprocessing, one for each of our broad general features: County age group populations, educational attainment rates, election data, race and sex combination populations, county unemployment rates, national unemployment rates, and national GDP growth and year over year change. The input for all notebooks were the raw data files from our sources. One notebook was used to merge the cleaned outputs of the seven aforementioned notebooks and produced our combined dataset. One notebook was used for cleaning data

to be used for machine learning. Two notebooks were used for data visualizations. One notebook was used for model preparation, training, and assessing models using sklearn. The modeling notebook is meant to be run entirely. It outputs plot figures and table results.

8. Members Participation

We decided that the optimal way to distribute work would be to allocate all the general responsibilities under a branch of data science to one person. While doing this, we still maintained communication in our group so we are all aware of where we were in our process, what we have left to do, and offer suggestions to each other. We believed allocating all the responsibilities under a single branch would allow us to somewhat specialize and focus our individual efforts. Constant communication was also important when we had directly related tasks such as preparing our comprehensive dataset which would be used in the machine learning portion of our project. We all contributed to preliminary data analysis.

Roy Minato	Finding data sources, pre-processing/wrangling and joining of data sources into our one comprehensive table, preliminary data analysis/visualizations
Shou Shimaya	Further cleaning the data so that it is ready to get fed into machine learning models. Did data visualization. Fitted logistic regression and did statistical tests.
Gurpal Kundi	Performed exploratory data analysis and data visualization. Conducted modeling and evaluation for machine learning models. Discussion and conclusion.

9. Discussion and Conclusion

- In terms of our modeling we learned about the different types of classification algorithms within the sklearn library. We learned about how to compare the accuracies of different models. However, we learned that these basic classification models are not well suited for time series problems. More sophisticated models are needed to deal with problems of this nature.
- The hardest part of this project was the data collection, wrangling, processing, and formatting for modeling. This took up the majority of our time. This was because we wanted our final dataset to incorporate many features. Data sets with these features had to be found, analyzed, and then processed. There was no single source, and the format of the raw data files were very different between sources, and even within sources.
- One aspect of our project that worked smoothly was division of responsibilities. We had initially not addressed this aspect of the project and had assumed that each member would contribute to each part of the pipeline. However, what ended up happening is that each part of the pipeline ended up being taken by each member. This allowed for each person to really get to know the details of their part of the pipeline, and aided in the efficiency of the project as a whole.
- One obvious future task would be to update our data set to include this year's data for the upcoming 2022 elections, and see how our model performs in real time. In terms of improving our model several ideas can be looked at. The first is modeling the problem as a longitudinal study and use models that deal with time dependency of the observations such as Recurrent Neural Networks (LSTM with PyTorch or Tensorflow) Generalized Linear Mixed-Effects Models (lme4 package in R). We could also look at incorporating other known dependencies such as spatial dependency between adjacent counties. We could Take into account X-factors during election years such as financial crises, pandemics, etc. We could also Include other types of elections.