Predicting History

Behind the Scenes of the Bloomberg News 2020 Voter Turnout Forecasts

Andrew Therriault, PhD

PyData Boston - February 23, 2021

andrewtherriault.com | twitter.com/therriaultphd | github.com/therriault

GETTING STARTED

MY BACKGROUND

- Political scientist by training, data scientist by trade, founder of Civin
- Led data science and analytics teams at the Democratic National Committee, City of Boston, and Facebook
- Taught data science grad courses at Harvard and Northeastern
- Co-organizer of PyData Boston chapter

WHY WE'RE HERE

- In advance of the 2020 election, I built a series of machine learning models to predict voter turnout for Bloomberg News to use in their election coverage
- We'll walk through the project and talk about the challenges and my thought process at each step

WHERE WE'RE GOING

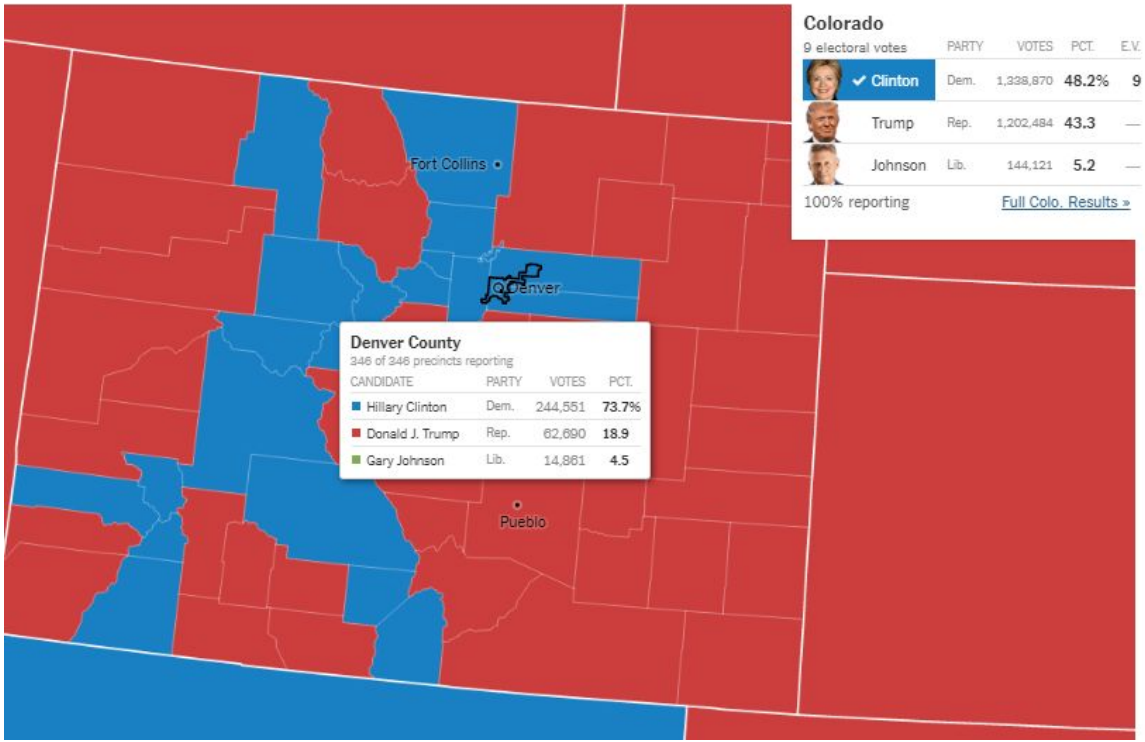
- Start by talking about the problem I was asked to solve
- Next, introduce the general design of my solution
- Go through some of the key details of implementation
- Recap how everything turned out and what we can learn

The Problem: Predicting Voter Turnout During a Pandemic

Basics of Election Night Coverage

- Election results are tabulated by local election officials at the precinct level and reported up to counties and states
- These results get reported to the Associated Press (AP) and other news services, who then publish combined tallies
- News organizations use the % of precincts reported as the key metric
- There are several complicating factors here:
    - Precincts are not all the same size, so 1% of precincts != 1% of votes
    - Early and absentee votes are often counted and reported at different times and levels
    - Specific procedures vary widely across cities, counties, and states
    - The order in which precincts report is highly non-random
- Nevertheless, % of precincts reported is generally a good approximation

What It Normally Looks Like



https://www.nytimes.com/elections/2016/results/president

The Pandemic Problem

- In 2020, 101.5 million votes cast before Election Day, out of 159.7 million
  - This equals 64% of total votes cast early (previous high was 34% in 2016)
  - 65.6 million by mail (41% of total), 35.8 million early-in-person (22% of total)
  - In many states, early votes were not allowed to be counted until polls closed, and processing mail-in ballots is generally slower than Election Day votes
  - In some states, early votes are counted at voters' local precincts, while in others they are handled at centralized facilities
  - Many (but not all!) states relaxed rules to make voting early easier, including allowing ballots mailed by Election Day to be received later
- With ⅔ of votes cast outside the normal polling places, the precincts-reported estimate is no longer reliable

My Assignment

- The AP promised to share overall state-level turnout predictions with member organizations, but by summer had not explained methodology and was not providing more granular data
- Bloomberg News reached out and asked me to provide forecasts…
  - For each state
  - For each county
  - For each type of election (Presidential, Senate, House, Governor)
- These forecasts would be used in place of precincts reported in their election coverage, to estimate the percentage of votes already counted

The Solution: Designing a Turnout Forecast Model

Some Fundamental Problems

- An election is a one-time event
  - It's ***very hard*** to predict something that has never happened before
  - It's ***impossible*** to quantify uncertainty when predicting a one-time event
- The typical approach to forecasting (using similar elections in the past) relies on having comparable data and data-generating mechanisms
  - County-level presidential election returns are only available going back to 2000, and other data sources (such as the American Community Survey) don't even go back that far
  - County and congressional district boundaries change over time
  - Rules around things like absentee, early-in-person, and all-mail voting are ever-changing
- 2020 was an ***extremely*** atypical year with no great precedent to train on

Finding a “Good Enough” Solution

- My basic philosophy for these projects:
  - The most important question isn’t “*is this perfect*?”, it’s “*is this better than the alternative*?”
  - Recognizing that your solution will never be perfect is very freeing
  - Start with a baseline answer, then focus on making your answers less wrong
  - Accept that your data will always be messy, just like everyone else’s
  - Be very honest about what you don’t or can’t know, and plan accordingly
- What this meant on this project:
  - Trading off the amount of data used vs. the accuracy and representativeness of that data
  - Using past turnout in each county & state as baselines, then adding election-specific info
  - Approximating uncertainty by testing the methodology on other elections
  - Making 3 distinct forecasts for different overall turnout level scenarios

The Final Model Design

- Models county-level presidential turnout rates (as a percentage of the citizen 18+ population) in 2004, 2008, 2012, and 2016 as a function of:
  - County- and state-level turnout in previous presidential and midterm elections
  - Voting regulations (early and absentee voting, felon disenfranchisement, voter ID, etc.) as well as changes therein since the previous election
  - Local population demographics (age, education, race, income, housing, employment, etc.)
  - Competitiveness of the races for president, senate, and governor in that state
- Model downballot “drop-off” from presidential turnout rates as a function of competitiveness, voting rules, and socioeconomic status
- Validate models by holding out specific elections to estimate uncertainty
- Retrain on all elections and predict for 2020 context, then make additional forecasts for “high” and “historic” turnout levels

Implementing the Forecast

Finding the Right Data

- Probably half the work in this project was just compiling data across years
- A partial list of specific sources:
  - Election results: mainly from the MIT Election Data and Science Lab, supplemented with Ballotpedia and Wikipedia as needed for specific cases
  - Demographic data: the Census Bureau's American Community Survey and Current Population Survey datasets, with some imputation pre-2010
  - Current and historical election competitiveness from FiveThirtyEight, Ballotpedia, and (mostly pre-2012) a bunch of other contemporary sources
  - Current and historical voting rules from the National Conference of State Legislatures, the Sentencing Project, the Pew Research Center and others
- Getting consistent data from one election to the next was a huge pain
- There's a lot of subjectivity in deciding when data is good enough to use

My Jupyter Notebook Workflow

- My normal workflow:
  - v0.1: Do everything in linear code in a notebook (or series of notebooks)
  - v0.2: Move generalizable things into functions and set parameters at the start
  - v0.3: Move functions into modules and import them into notebook
  - v0.4 (*optional*): Move the code that calls those functions into a standard Python script, and maybe also move the parameters into a JSON file
  - v1.0: Clean out everything that's not essential and put into production
- My workflow on this project: hybrid of v0.1 (for brand new code) and v0.3 (for code adapted from other projects)
- Not the most elegant approach, but it worked pretty well here (wouldn't recommend it for collaborative or repeated projects, though!)

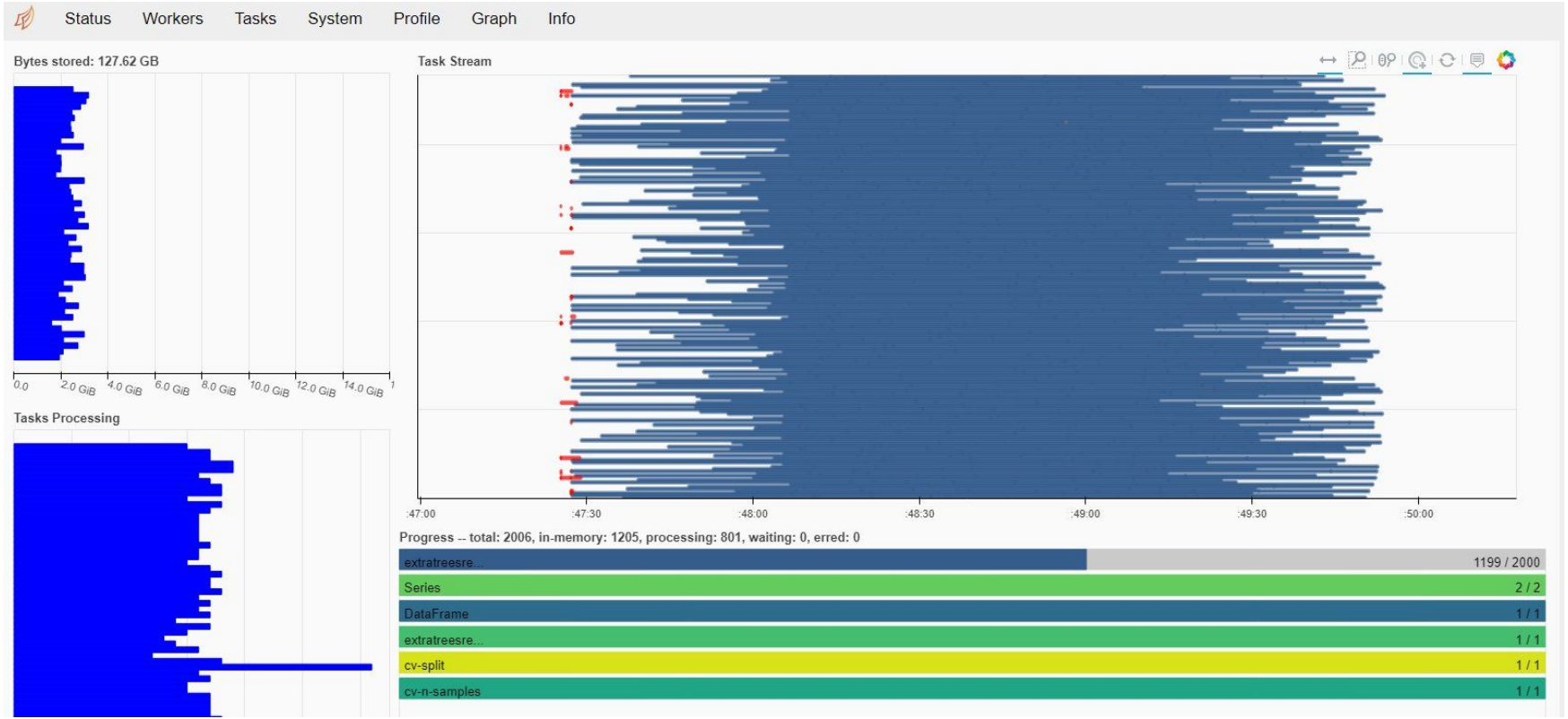
Creating Ensembles Across Models and Algorithms

- For modeling presidential turnout rates, we used three different tree-based algorithms (all implemented in scikit-learn):
  - Random Forest Regression
  - Extra Trees Regression
  - Gradient-Boosted Decision Tree Regression
- Rationale: each algorithm handles features in a different way, so combining them gives each feature a chance to have an impact
- Also tried linear regression and regularized linear regressions (lasso, ridge, elastic net), but they actually made the ensemble predictions worse
- Did use linear regression for downballot models though, as those had simpler specifications and much smaller samples

High-Performance Model Tuning with Dask

- With so much work collecting and refining the data used, a lot of time was spent tuning and re-training models and improving feature selection
- 3000+ counties * 4 years * 50-100 features * 3-5 algorithms * 3-5 hyperparams per algo * (4 validations + 1 final model) = a lot of compute!
- My laptop = lightweight and nice to look at, but not that powerful
- Solution: parallelization with Dask, deployed on the Coiled Cloud platform
    - Dask is a drop-in replacement for joblib in scikit-learn - instead of just parallelizing across one machine, you can parallelize across a cluster
    - Change from joblib to Dask back-end took < 10 lines of code and under an hour to set up
    - Coiled platform (from Dask's creators) provides a hosted Dask cluster as a service on AWS
    - As part of Coiled Beta, I was able to use up to 200 cores on their cluster, and I ended up distributing tuning across 48 nodes with 4 cores each

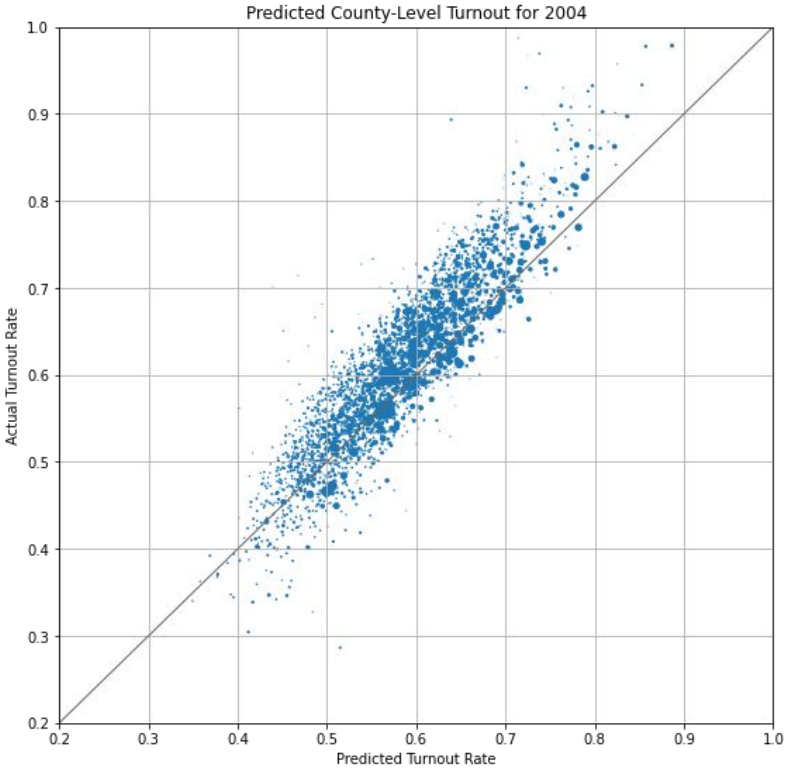
10-fold CV x 5 algos x 200 hyperparameter combos

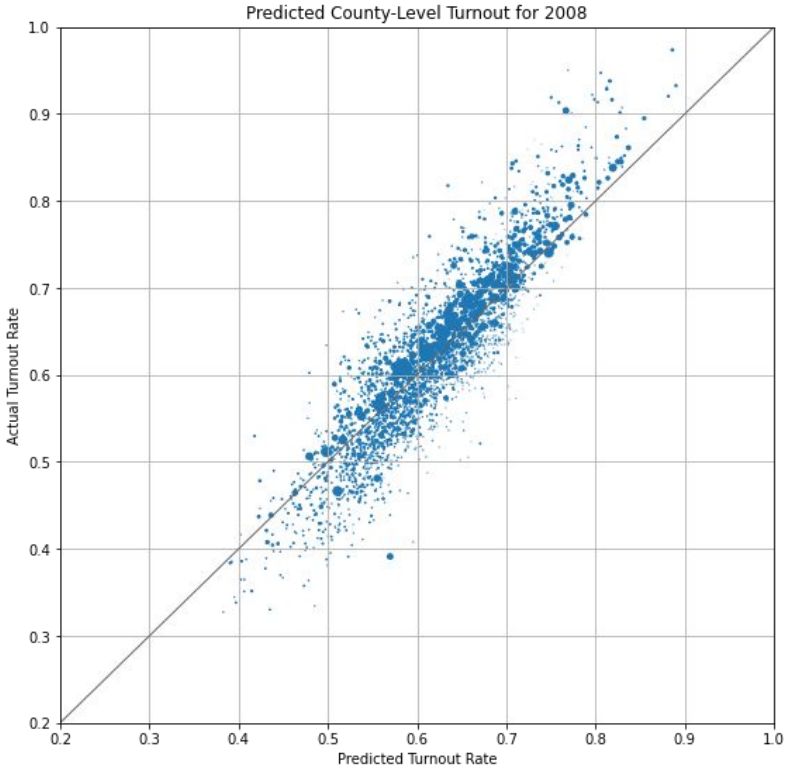


Validating Models and Estimating Uncertainty

- To validate presidential models, we did 4 runs, each holding out 1 year and using it for testing
- This gave us a sense of both how accurate our predictions were across states and counties, as well as how the year-by year bias varied
- Across the 4 years used, our total national vote forecasts ranged from 3.5% too low (in 2004) to 6.3% too high (in 2012)
- At the county level:
    - The 10th percentile difference between predicted and actual turnout was -5.3%
    - The 90th percentile was +6.0%
    - The median absolute error for a given county was +/- 3.1% from our forecast

Validating Models and Estimating Uncertainty





Getting Ready for Release

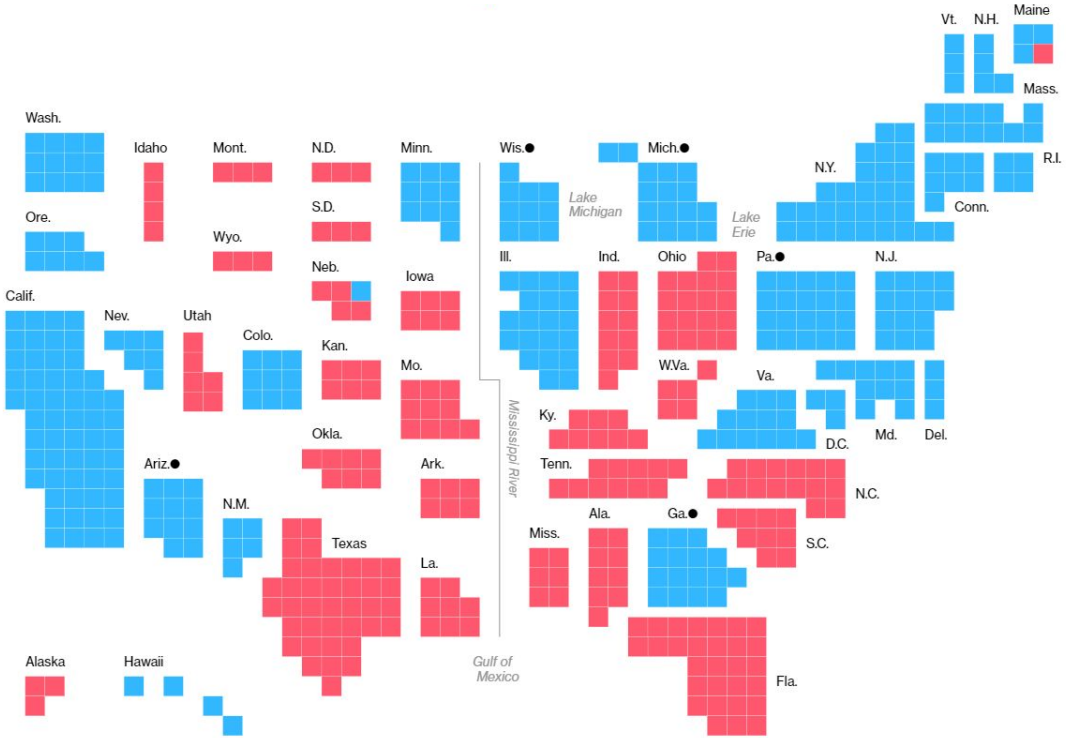
- We decided to present our predictions as a range of potential vote counts (+/- 5% of our forecast) rather than a single number
- This range was applied at all levels (which allows for correlated errors)
- Knowing that we couldn't fully model what was likely to be a very high turnout election, we generated three forecasts:
    - ***Baseline turnout***: our model's predicted turnout level
    - ***High turnout***: baseline turnout + 5%
    - ***Historic turnout***: high turnout + 5%
- We also wrote up an extensive methodology page and published it on the site the week before

What Happened?

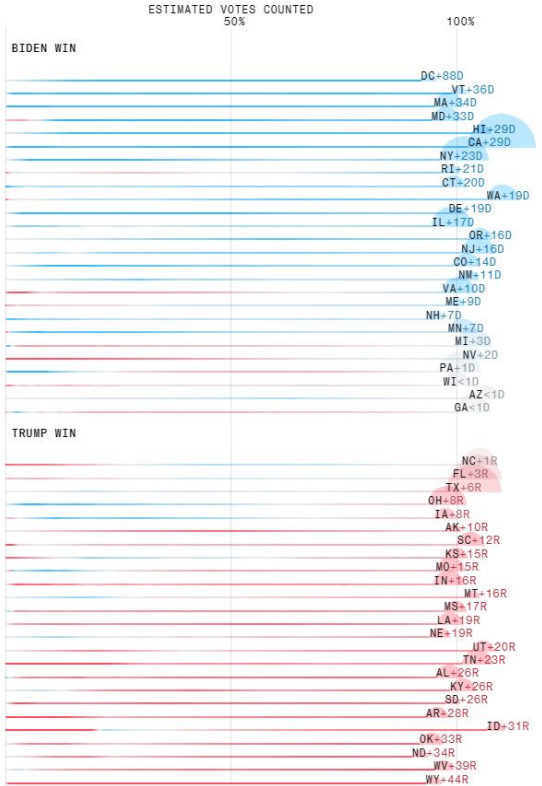
The Plan for Election Night

- On Election Day, we'd load up our baseline forecast on the site, with every county and state showing a range of total votes expected
- We set conditions for changing from the baseline to high turnout and historic turnout forecasts as votes came in, based on a minimum number of counties / states reporting turnout above our expected range
- The site would show counting progress as a percentage of the expected number of votes under the current forecast, and also note which forecast those estimates are based on

National Tracker

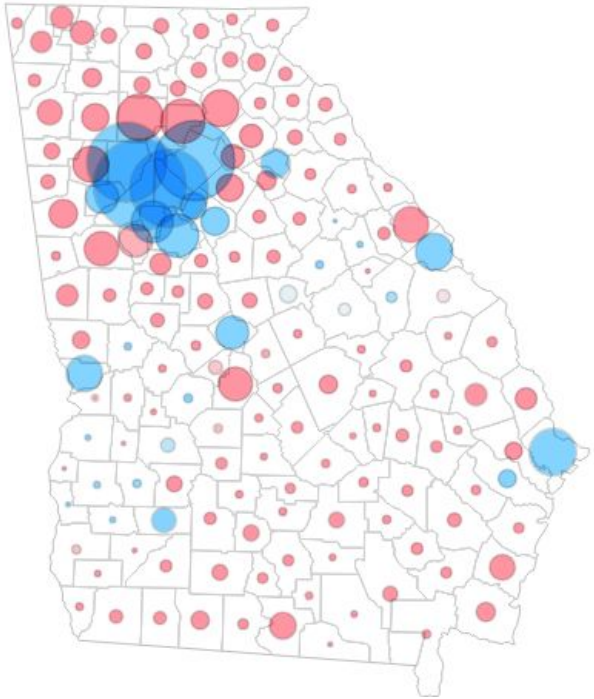


Note: In addition to awarding electoral votes to the winner statewide, Maine and Nebraska award an elector to the winner in each congressional district.



https://www.bloomberg.com/graphics/2020-us-election-results

State Tracker



Presidential Results

	VOTES	PCT.
Joe Biden ✓ DEM	2,473,633	49.5%
Donald Trump * REP	2,461,854	49.3%
Jo Jorgensen LIB	62,229	1.2%

*INCUMBENT

PAST THREE ELECTIONS
2016 R 2012 R 2008 R

How many votes have been counted?
5.00M out of 4.78-5.28M estimated total votes

95-105%
of est. total votes

D+0.24%

0% 25% 50% 75% 100%

Total vote estimates assume **historic** voter turnout.
Turnout estimates have a margin of error of +/-5%.

https://www.bloomberg.com/graphics/2020-us-election-results/georgia

The Problem with Florida

- Florida reported ***super*** early - most counties were complete or close to it by around 9pm
- As a result, our model was still set to “baseline” turnout, and we were seeing turnout numbers coming in as high as 120%+ of the forecast
- We knew at that point that our baseline forecast was too low, but because Florida was so early, we hadn’t met the trigger for going to “high” turnout
- We ultimately did switch the forecast to “high” around midnight, and then to “historic” around 5am

The Problem with New York

NEW YORK STATE

What can lawmakers do about NY's maddeningly slow vote count?

The long, drawn-out absentee tally is a national embarrassment.

By ANNIE MCDONOUGH | DECEMBER 6, 2020

SHARE:

To say that New York has been slow to count absentee ballots this year is an understatement. By the time enough absentee ballots were counted to allow Democratic state Senate leaders to celebrate picking up two new seats to win a supermajority, the entire state of Georgia had already counted and certified its votes twice. Meanwhile in New York, some local boards of election missed the Nov. 28 legal deadline to submit final election results to the state.

The counting of absentee ballots didn't even begin in New York until at least a week after the election – some local boards of election waited even longer. And it wasn't just November's election results that were delayed. The winners of two congressional primary races this year weren't declared until six weeks after the June election.

https://www.cityandstateny.com/articles/politics/new-york-state/what-can-lawmakers-do-about-nys-maddeningly-slow-vote-count.html

Overall Performance

- Nationally, our model's historic forecast was right on target: the final count was just shy of 160 million, and our forecast was 161 million
- In key swing states, the turnout relative to our historic forecast was incredibly close:
    - Georgia: 100%
    - Michigan: 100%
    - Nevada: 101%
    - North Carolina: 101%
    - Pennsylvania: 97%
    - Wisconsin: 97%
- Our forecasts were a little less accurate in some non-swing states like Hawaii and Idaho, but all states fell within +/- 7% of the historic forecast, and most were within +/- 5% as intended

Final Takeaways

- Overall, the model worked surprisingly well under the circumstances
- This is mainly due to the humility of our approach, especially:
  - Presenting forecasts as ranges rather than specific targets
  - Not assuming that errors will average out across predictions (as poll aggregators often do)
  - Accepting that some things were impossible to know in advance, and designing a system (by producing three forecasts) that allows us to adapt to new incoming information
- What could be improved:
  - More attention paid to data quality (which might mean scaling back ambitions)
  - Be willing to take greater risks with the adaptive forecast (which reacted too slowly)
  - Confirm whether all the complexity is really necessary

Thank you!

Find me on Twitter @therriaultphd or email andrew@civin.co.
Slides are available at https://github.com/therriault/slides.